Conf. dr. ing. Adrian Popia
Ş.l. dr. ing. Rodica Popia

CALCULUL ŞI COMPENSAREA REȚELELOR DE TRILATERAȚIE

CUPRINS

Cap. 1 Elemente generale
1.1 Notiuni privind masurarea electronica a distantelor..... 1
1.2 Definirea notiunii de trilateratie..... 4
1.3 Caracteristicile retelelor de trilateratie..... 4

Cap. 2 Executia masuratorilor în retelele de trilateratie
2.1 Tipuri de aparate de masura si caracteristici ale acestora..... 5
2.2 Realizarea masuratorilor în retelele de trilateratie..... 6

Cap. 3 Calculul si compensarea retelelor de trilateratie prin metoda masuratorilor indirecte ponderate
3.1 Calcule preliminare..... 7
3.1.1 Reducerea matematica a distantelor masurate..... 7
3.1.2 Reducerea distantelor la planul proiectiei..... 9
3.1.3 Calculul coordonatelor provizorii ale punctelor noi..... 10
3.1.4 Calculul distantelor si orientarilor provizorii în retea..... 13
3.2 Problema ponderii masuratorilor în retelele de trilateratie..... 13
3.3 Compensarea retelelor de trilateratie prin metoda masuratorilor indirecte ponderate..... 14
3.3.1 Aspecte ale ecuatiilor de corectii..... 14
3.3.2 Calculul coeficientilor ecuatiilor de corectii..... 15
3.3.3 Formarea matricii ecuatiilor initiale de corectii..... 16
3.3.4 Formarea matricii ecuatiilor transformate echivalente de corectii..... 17
3.3.5 Crearea sistemului liniar al ecuatiilor normale ale corectiilor coordonatelor punctelor noi..... 19
3.3.6 Calculul coordonatelor compensate ale punctelor noi..... 21
3.3.7 Calculul distantelor compensate..... 21
3.3.8 Evaluarea preciziei rezultatelor..... 22

Cap. 4 Exemplu de calculul si compensare a unei retele de trilateratie
4.1 Datele de baza 23
4.2 Calcule preliminarii..... 24
4.3 Calculul coeficientilor si termenilor liberi ai ecuatiilor initiale de corectii..... 25
4.4 Calculul ponderilor ecuatiilor initiale de corectii..... 25
4.5 Formarea matricii ecuatiilor initiale de corectii si a matricii

4.8 Calculul coordonatelor compensate ale punctelor noi..... 32
4.9 Calculul corectiilor distantelor masurate si al distantelor compensate..... 32
4.10 Evaluarea preciziei rezultatelor..... 34

Cap. 5 Schema logica generala si program în limbaj Borland Pascal 7.0, pentru calculul si compensarea retelelor de trilateratie
5.1 Prezentare..... 35
5.2 Textul programului sursa..... 37
5.3 Exemplu de calcul rezolvat cu programul prezentat..... 98

Bibliografie..... 108

Cap. 1 Elemente generale

1.1 Noțiuni privind măsurarea electronică a distanțelor

Se poate aprecia că fazele precursoare ale dezvoltării tehnicii de măsurare electronică a distanțelor pot fi considerate descoperirea în 1890 a celulei Kerr şi punerea la punct a radiolocației în perioada 1930-1940.

În principiu, măsurarea electronică a distanțelor se bazează pe emisia şi recepția undelor electromagnetice din diferite spectre de frecvență, mai precis pe determinarea timpului de propagare a undelor în lungul distanței măsurate. Valoarea distanței rezultă indirect, ca un produs între timpul de propagare şi viteza de deplasare a undei, egală cu viteza luminii.

Printre aplicațiile civile ale acestui principiu de măsurare, cele din domeniul geodeziei au o mare importanță, în sensul creşterii preciziei de determinare spațială a detaliilor scoarței terestre, cu implicații directe în foarte multe domenii ale activității umane.

Aplicațiile din geodezie ale măsurării electronice a distanțelor au permis dezvoltarea unor sisteme de înaltă tehnologie precum sistemul satelitar de poziționare globală, denumit prescurtat GPS şi aparatele topografice electronice, cum sunt stațiile totale.

Dispozitivele electronice pentru măsurarea distanțelor, utilizate în domeniul geodeziei sunt de două tipuri, în funcție de spectrul de frecvență al semnalului utilizat:

- telemetre electrooptice, pe bază de unde electromagnetice din spectrul luminii;
- radiotelemetre, bazate pe unde electromagnetice din spectrul radio.

În funcție de modul de determinare a timpului de propagare, aceste dispozitive sunt:

- cu funcționare în impulsuri(principiul radarului);
- cu funcționare pe bază de diferență de fază.

De-a lungul timpului, telemetrele pe bază de impuls au fost puțin utilizate datorită preciziei relativ scăzute, utilizându-se în schimb cele de tip electrooptic fazic. Acestea din urmă permit determinarea distanțelor după relația generală:

unde *c* este viteza de propagare a luminii în aer; *w* - pulsaţia undei electromagnetice modulatoare; *j* - faza undei modulatoare în momentul recepţiei.

Pornind de la relaţia de mai sus sistemul de poziţionare globală(GPS) exploatează modul de măsurare în timp absolut, iar staţiile totale modul de măsurare în timp relativ.

Deoarece indicatorul de fază al dispozitivelor electronice utilizate de staţiile totale dă numai diferenţa de fază în limitele ultimei perioade de oscilaţie (de variaţie a fazei) modulatoarei, iar în timpul de parcurs dus-întors al undelor pot avea loc N perioade complete de oscilaţie, distanţa se va determina cu relaţia:

$$D = \frac{1}{2} \cdot \lambda \left(N + \frac{\Delta j}{2\pi} \right) \qquad (1.2)$$

unde λ este lungimea de undă a modulatoarei în aer; Δj -diferenţa de fază în ultima perioadă a timpului de parcurs şi *N*-numărul de perioade complete de variaţie a fazei produse pe parcursul dus-întors. Determinarea numărului, *N* de perioade complete se realizează prin numărarea efectivă a trecerilor prin zero ale semnalului modulator între momentul începerii emisiei şi momentul recepţiei semnalului reflectat. Obţinerea unei precizii acceptabile a rezultatului măsurării se realizează prin medierea rezultatelor obţinute din repetarea automată a determinării, atunci cînd aparatul foloseşte o singură frecvenţă de modulaţie sau din utilizarea a două sau mai multe frecvenţe de modulaţie.

În general, la dispozitivele electronice pentru măsurarea distanţelor din staţiile totale se utilizează ca undă purtătoare lumina infraroşie, cu o frecvenţă de circa 10^{12}-10^{14} Hz(lungimi de undă de circa 0.76-10μm) şi ca undă modulatoare o undă radio de tip metric, cu frecvenţa de circa 6-30 10^9 Hz(lungimi de undă de circa 10-50m). Ţinând cont de aceste elemente, numărul *N* de perioade întregi contorizat de dispozitiv, poate varia între zero şi câteva sute, pentru distanţe măsurate în ecartul 0.5-10000 m

Rezultatul brut al măsurării, tradus în distanţă înclinată, este supus în general procesului de corecţie fizică, de temperatură şi presiune atmosferică, elemente care influenţează propagarea undelor electromagnetice în aer.

Procesul de corecție matematică, aplicat automat în stația totală asupra distanței înclinate corectată fizic se referă la efectul de curbură a Pământului şi de refracție atmosferică şi, la reducerea la orizont a acesteia.

În general, într-o stație totală(de exemplu RecElta 3), distanța înclinată, căreia i s-a aplicat corecția pentru factorii meteorologici, L_k, va rezulta după măsurarea distanței înclinate brute(necorectate), L_{br} sub forma:

$$L_k = \left(L_{br} + L_{\text{int}} + L_{pr}\right) \times c_f \tag{1.3}$$

unde L_{pr} este constanta adițională a prismei reflectoare, L_{int}-corecția internă a dispozitivului electronic de măsurare a distanței, iar factorul de corecție fizică, c_f se calculează cu:

$$c_f = \left[1 + \left(n_0 - n\right) \times 10^{-6}\right] \times \left(1 + a \times T^2 \times 10^{-6}\right) \tag{1.4}$$

în care:

n_0*=255* -este indicele de refracție de bază;

$n = \left(79.146 \times P\right)/\left(272.479 + T\right)$ -este indicele de refracție curent;

a=0.001-este coeficientul de corecție pentru presiunea vaporilor;

T-este temperatura curentă, în ^{0}C;

P-este presiunea atmosferică curentă, în hPa(mbar).

Prin corecția matematică a distanței L_k rezultă distanța redusă la planul orizontal, D_k, sub forma:

$$D_k = \left(D_1 + D_2\right) \times M \tag{1.5}$$

unde:

$D_1 = L_k \times \sin\left(V + R\right)$;

V –este unghiul vertical zenital măsurat pe aliniamentul respectiv;

$R = 6.5 \times 10^{-7} \times L_k \times \sin\left(V\right)$ este efectul refracției atmosferice;

$D_2 = -1.57 \times 10^{-8} \times \Delta h \times L_k \times \sin\left(V\right)$ este efectul curburii Pământului;

Δh – diferența de nivel între capetele distanței măsurate;

M –este coeficientul de scară pentru planul la care se reduce distanța; de

1.2 Definirea noțiunii de trilaterație

Se ştie că reţelele planimetrice de sprijin sunt formate din puncte, care unite între ele cu linii imaginare formează o serie de triunghiuri alăturate. Pentru determinarea coordonatelor punctelor reţelei se poate opta fie pentru măsurarea tuturor unghiurilor din triunghiuri şi a unor laturi pentru verificare(măsurători de triangulație), fie pentru măsurarea tuturor laturilor reţelei şi a unor unghiuri pentru verificare(măsurători de trilaterație), fie pentru ambele tipuri de măsurători simultan.

În consecință, trilaterația este procesul de măsurare a distanțelor (laturilor) în reţelele planimetrice de sprijin cu scopul determinării coordonatelor punctelor din aceste reţele.

Deoarece aparatura electronică de măsurare a distanțelor oferă în momentul de față precizii de măsurare foarte bune şi, realizarea măsurătorilor liniare este mult mai uşoară decât cea a măsurătorilor unghiulare, trilaterația se poate considera ca una din metodele foarte economice de creare a reţelelor planimetrice de sprijin.

1.3 Caracteristicile reţelelor de trilaterație

Execuția trilaterației presupune ca toate punctele reţelei să fie accesibile deoarece la fiecare latură măsurată, într-un capăt se instalează instrumentul de măsură iar în celălalt capăt se instalează reflectorul.

Prin măsurători de trilaterație se pot realiza reţele de puncte de sprijin noi sau se poate reabilita şi îndesi o reţea de triangulație mai veche, cu condiția de a se folosi doar punctele accesibile ale respectivei reţele.

Ținând cont de ecartul de distanțe măsurat de aparatura electronică curentă (0.001-10 km) rezultă că se poate lucra în reţele de ordinele III-V, cu foarte bune performanțe, în condițiile în care se respectă criteriile de vizibilitate reciprocă a punctelor, la fel ca şi la triangulație.

Trebuie amintit că există aparatură electronică de măsură şi pentru reţele de ordin superior (cu posibilitate de măsurare a distanțelor de până la 80 km), dar aceasta este mai puțin utilizată odată cu dezvoltarea tehnologiei GPS.

Cap. 2 Execuția măsurătorilor în rețelele de trilaterație

2.1 Tipuri de aparate de măsură şi caracteristici ale acestora

Alegerea aparaturii de măsură în trilaterație se face în funcție de ordinul rețelei, deci în concordanță cu lungimile maxime ale laturilor.

Pentru rețele de ordin superior se utilizează telemetre electrooptice sau radiotelemetre cu rază mare de acțiune cum sunt cele din tabelul 2.1

Tab. 2.1 Aparate cu rază mare de acțiune pentru măsurarea distanțelor

Marcă	Tip aparat	Rază de acțiune(km)	Eroare medie(cm)
PEM 2	radiotelemetru	0.1-70	$\pm\left(3+3\cdot10^{-6}\cdot D^{*}\right)$
DM 20	telemetru electrooptic	0.07-80	$\pm\left(1+3\cdot10^{-6}\cdot D\right)$
DI 50	distomat	0.1-50	$\pm\left(2+10^{-6}\cdot D\right)$

D*-distanța măsurată

Pentru rețele de ordin inferior pot fi utilizate telemetre specializate sau stații totale ca de exemplu cele din tabelul 2.2

Tab. 2.2 Aparate cu rază de acțiune medie pentru măsurarea distanțelor

Marcă	Tip aparat	Rază de acțiune cu o prismă (km)	Eroare medie
RecElta 13C	stație totală	2	$\pm\left(2mm+2ppm\right)$
RecElta 15C	“	1.5	$\pm\left(3mm+2ppm\right)$
Elta R 45	“	1.5	$\pm\left(5mm+3ppm\right)$
Elta S10	“	2.5	$\pm\left(1mm+2ppm\right)$
Elta S20	“	2.5	$\pm\left(2mm+2ppm\right)$
Leica TC 605-905	“	3.5-9.0	$\pm\left(2mm+2ppm\right)$
Leica TC 1010-1610	“	3.5-9.0	$\pm\left(2mm+2ppm\right)$
Geodimeter 640	“	2.5	$\pm\left(3mm+3ppm\right)$
Topcon GTS 710-713	“	2.4	$\pm\left(2mm+2ppm\right)$

2.2 Realizarea măsurătorilor în rețelele de trilaterație

În rețelele de trilaterație se staționează în general toate punctele. Pentru operativitate este bine să se lucreze cu mai multe prisme de acelaşi tip, astfel încât să fie acoperite simultan toate laturile măsurate din acelaşi punct. Este recomandabil ca atât instrumentul cât şi reflectoarele să fie instalate centric, prin centrare forțată, cu erori de valori foarte mici(sub 1 mm). Multe din aparatele de măsură utilizate permit atât determinarea distanțelor înclinate şi a unghiurilor verticale cât şi reducerea directă a distanțelor la orizont sau chiar la planul proiecției în care se lucrează. În plus stațiile totale aplică automat atât corecțiile fizice de temperatură şi presiune, valori care sunt măsurate prin intermediul senzorilor proprii ai aparatelor sau măsurate separat de către operator şi introduse în memoria stației totale înainte de măsurători, cât şi corecțiile matematice de curbură, de refracție atmosferică şi de reducere la planul proiecției.

Măsurătorile de laturi se recomandă să fie realizate cu minimum 3 serii complete de observații, iar în calculul de compensare să se utilizeze valorile medii. De obicei laturile sunt măsurate în ambele sensuri, astfel că în rețea există măsurători în surplus, ceea ce permite o compensare riguroasă a rețelei prin metoda micilor pătrate.

Cap. 3 Calculul şi compensarea reţelelor de trilaterație prin metoda măsurătorilor indirecte ponderate

3.1 Calcule preliminare

Prelucrarea preliminară a observațiilor efectuate în reţelele de trilaterație constă în general din:

- corecția (reducerea) fizică a distanţelor măsurate, în funcție de factorii atmosferici, temperatură şi presiune în momentul măsurării;
- corecția (reducerea) matematică a distanţelor măsurate, care include corecțiile pentru curbura fasciculului luminos, pentru verticalitatea axului principal al aparatului, pentru reducerea la orizont şi pentru reducerea la elipsoidul de referință;

La staţiile totale corecția fizică, dar şi corecțiile de curbură a fasciculului, de verticalitate a axului principal şi de reducere la orizont se aplică automat.

Pentru aplicarea corecției fizice reale, operatorul va trebui să măsoare şi să introducă valorile reale ale temperaturii şi presiunii, la staţiile totale care nu sunt prevăzute cu senzori proprii pentru măsurarea valorilor respective.

3.1.1 Reducerea matematică a distanţelor măsurate

În cazul în care corecțiile matematice se aplică manual, aceste operații se realizează după relațiile şi, în ordinea de mai jos:

a) Corecția de curbură a fasciculului luminos(*corecția de refracție atmosferică*), DD_c se calculează cu relația:

$$\Delta D_c = k^2 \times \left(\frac{L_i^3}{24 \times R_0^2} \right) \qquad (3.1)$$

unde $k=0.13$ este coeficientul de refracție; L_i – distanța înclinată corectată fizic şi $R_0 = 6368 \cdot 10^3$ m este raza medie de curbură a sferei terestre.

b) Corecția de verticalitate a axului principal al aparatului, DD_v se determină

unde *d* este înălţimea axei optice a aparatului, deasupra planului de referinţă şi a_v-eroarea de verticalitate a axului principal.

c) *Corecţia de reducere la orizont, DD_0* se calculează cu relaţia:

$$\Delta D_0 = -\frac{dh^2}{2 \times D_i} - \frac{dh^4}{8 \times D_i^3} = -\frac{(H_2 - H_1)}{2 \times D_i} - \frac{(H_2 - H_1)}{8 \times D_i^3} \qquad (3.3)$$

unde H_2 şi H_1 sunt altitudinile în cele două capete ale distanţei măsurate şi D_I – distanţa înclinată măsurată

Ţinând cont de relaţiile 3.1-3.3, distanţa redusă la orizont, D_0 va fi:

$$D_0 = D_i + \Delta D_c + \Delta D_v + \Delta D_0 \qquad (3.4)$$

iar *distanţa redusă la elipsoidul de referinţă* se va calcula cu relaţia:

$$D_e = D_0 + \Delta D_e \qquad (3.5)$$

unde corecţia de reducere la elipsoid, ΔD_e se calculează cu:

$$\Delta D_e = -\frac{dh^2}{2 \times D_0} - \frac{dh^4}{8 \times D_0^3} - \frac{H_m}{N_1} \times D_0 + \\ + \frac{dh^2 \times H_m}{2 \times N_1 \times D_0} + \frac{D_0^3}{24 \times N_1^3} - \frac{H_2}{N_1} \times D_0 \times h^2 \times \cos^2 A_{12} \qquad (3.6)$$

în care $dh=H_2-H_1$ este diferenţa de nivel între capetele laturii măsurate; $H_m = \frac{H_1 + H_2}{2}$ este cota medie a laturii; N_1- raza de curbură a primului vertical în punctul de staţie; A_{12}-azimutul laturii şi $h = e' \cdot \cos j = 34 \cdot 10^{-4}$, pentru latitudinea medie a ţării.

În cazul reţelelor de ordin inferior reducerea distanţelor înclinate la orizont şi la elipsoidul de referinţă se realizează conform relaţiilor:

$$D_e = D_0 \cdot \left(1 - \frac{H_m}{R_0}\right) \tag{3.8}$$

în care v_z este unghiul vertical zenital al aliniamentului respectiv şi ceilalţi termeni au fost explicaţi mai sus.

3.1.2 Reducerea distanţelor la planul proiecţiei

Valoarea distanţei în planul proiecţiei, D_{pr} se calculează cu relaţia:

$$D_{pr} = D_e \cdot \left(1 + k_{pr}\right) \tag{3.9}$$

unde D_e este distanţa redusă pe elipsoidul de referinţă şi k_{pr}-coeficientul de corecţie corespunzător planului de proiecţie adoptat.

Pentru planul proiecţiei Gauss, coeficientul k_{pr} se calculează cu:

$$k_{pr} = \frac{Y_m^2}{2 \cdot R_m^2} + \frac{\Delta Y^2}{24 \cdot R_m^2} - \frac{Y_m^4}{12 \cdot R_m^2} \tag{3.10}$$

unde $R_m = \sqrt{M \cdot N}$ este raza medie de curbură a elipsoidului, corespunzătoare latitudinii mijlocului distanţei măsurate(M şi N sunt razele de curbură ale secţiunilor normale principale în punctul considerat); Y_m-ordonata mijlocului laturii măsurate şi DY-diferenţa de ordonate ale capetelor laturii măsurate.

Pentru planul proiecţiei Stereo70, coeficientul k_{pr} se calculează cu relaţia:

$$k_{pr} = 0.99975 \cdot \left(\frac{X_m^2 + Y_m^2}{4 \cdot R_0^2} - \frac{X_m^2 + Y_m^2}{48 \cdot R_0^2}\right) \tag{3.11}$$

unde $X = \frac{X_1 + X_2}{2} - X_0$ şi $Y = \frac{Y_1 + Y_2}{2} - Y_0$ sunt coordonatele

3.1.3 Calculul coordonatelor provizorii ale punctelor noi

Deoarece în rețeaua de trilaterație elementele cunoscute inițial sunt coordonatele punctelor vechi şi laturile măsurate, coordonatele provizorii ale punctelor noi, X_i^0 şi Y_i^0 vor fi calculate în funcție de aceste elemente. În general, poziția unui punct nou, $P(x_p,y_p)$ este determinată univoc dacă se dispune de un set de 3 distanțe măsurate între acest punct şi 3 puncte de coordonate cunoscute; doar 2 puncte cunoscute sunt necesare pentru determinarea coordonatelor punctului nou, iar al treilea punct cunoscut permite verificarea determinării.

Presupunem punctele cunoscute $1(x_1,y_1)$, $2(x_2,y_2)$ şi $3(x_3,y_3)$, punctul nou $P(x_p,y_p)$ şi distanțele reduse la planul proiecției $D_{1\text{-}p}$, $D_{2\text{-}p}$, şi $D_{3\text{-}p}$, rezultate din măsurători(fig. 3.1).

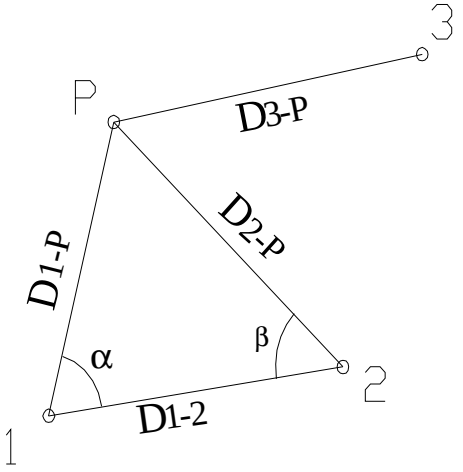

Fig. 3.1 Calculul coordonatelor provizorii ale punctului nou prin intersecția a două distanțe

În triunghiul *1-2-P*, latura $D_{1\text{-}2}$ poate fi determinată cu coordonatele cunoscute ale punctelor *1* şi *2*:

iar laturile D_{1-p} şi D_{2-p} sunt cunoscute din măsurători. Cu aceste elemente se calculează semiperimetrul, S cu relaţia:

$$S = \frac{1}{2} \cdot \left(D_{1-2} + D_{1-p} + D_{2-p} \right) \tag{3.13}$$

Unghiurile a şi b se pot determina cu relaţiile:

$$a = 2 \cdot arctg \left(\frac{\sqrt{\left(S - D_{1-p}\right) \cdot \left(S - D_{1-2}\right)}}{\sqrt{S \cdot \left(S - D_{2-p}\right)}} \right)$$

şi (3.14)

$$b = 2 \cdot arctg \left(\frac{\sqrt{\left(S - D_{2-p}\right) \cdot \left(S - D_{1-2}\right)}}{\sqrt{S \cdot \left(S - D_{1-p}\right)}} \right)$$

În continuare, coordonatele punctului nou, P se pot determina prin metoda intersecţiei înainte procedeul unghiurilor, prin metoda radierilor sau prin intersecţie de distanţe:

a) Calculul coordonatelor punctului nou prin intersecţie unghiulară înainte

Coordonatele x_p şi y_p ale punctului nou se calculează cu relaţiile:

$$x_p = \frac{x_1 \cdot tga + x_2 \cdot tgb \pm \left(y_2 - y_1\right) \cdot tga \cdot tgb}{tga + tgb}$$

$$y_p = \frac{y_1 \cdot tga + y_2 \cdot tgb \mp \left(x_1 - x_2\right) \cdot tga \cdot tgb}{tga + tgb} \tag{3.15}$$

Se observă că al treilea termen al numărătorului în relaţiile (3.15) poate avea fie semnul +, fie semnul - , deci pot apare două soluţii, în funcţie de poziţia punctului nou faţă de latura formată de punctele *1* şi *2*. Din acest motiv se calculează prima soluţie, prin alegerea semnelor + în prima relaţie şi – în a doua şi, cu coordonatele obţinute se calculează distanţa până la punctul cunoscut 3, D_{3-p}, comparându-se cu valoarea obţinută la măsurători

b) Calculul coordonatelor punctului nou prin metoda radierilor

Orientarea laturii 1-2 , q_{1-2} , se poate determina din coordonate cu:

$$q_{1-2} = arctg\frac{y_2 - y_1}{x_2 - x_1} + k \cdot p \qquad (3.16)$$

Cu unghiurile *a* şi *b* se pot calcula orientările:

$$q'_{1-p} = q_{1-2} - a \quad \text{şi} \quad q'_{2-p} = q_{1-2} + 200 + b \qquad (3.17)$$

sau orientările:

$$q''_{1-p} = q_{1-2} + a \quad \text{şi} \quad q''_{2-p} = q_{1-2} + 200 - b \qquad (3.18)$$

Coordonatele punctului nou se pot calcula prin radiere din punctele *1* şi *2* , în două soluții , alegând fie orientările din relațiile (3.17) , fie pe cele din (3.18). Pentru prima soluție rezultă:

$$x_p = x_1 + D_{1-p} \cdot \cos q'_{1-p} \quad \text{şi} \quad x_p = x2 + D_{2-p} \cdot \cos q'_{2-p}$$

$$y_p = y_1 + D_{1-p} \cdot \sin q'_{1-p} \quad \text{şi} \quad y_p = y2 + D_{2-p} \cdot \sin q'_{2-p} \qquad (3.19)$$

Cu valorile din prima soluție se calculează distanța D_{3-p} , comparându-se cu cea din măsurători; în caz de neconcordanță, în relațiile (3.19) se vor utiliza orientările din varianta (3.18).

c) Calculul coordonatelor punctului nou prin intersecție a trei distanțe

Punctul nou se poate determina şi prin folosirea exclusivă a distanțelor. În principiu, un punct este determinat univoc prin măsurarea distanțelor de la acesta la trei puncte cunoscute. Se consideră punctele cunoscute *1, 2, 3* şi cercurile cu centrele în aceste puncte, de raze egale cu distanțele măsurate până la punctul nou, *P*. În sistemul rectangular plan, translat cu originea în punctul 1, ecuațiile celor trei cercuri sunt (fig. 3.2):

$$(x')^2 + (y')^2 - D^2_{1-p} = 0$$

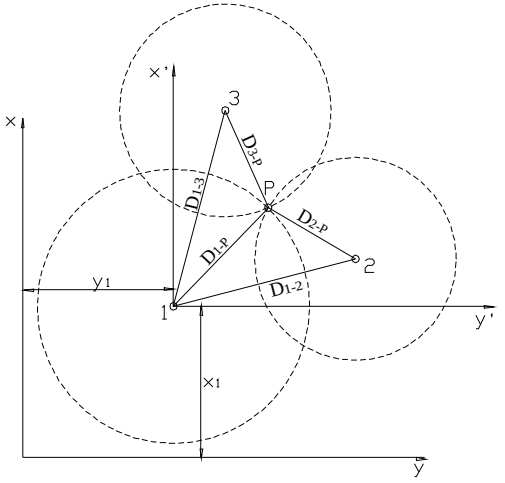

Fig. 3.2 Determinarea unui punct prin intersecție cu distanțe

Prin rezolvarea sistemului (3.19a) rezultă coordonatele $x_P^{'}$ şi $y_P^{'}$ în sistemul translat, după care coordonatele în sistemul general de axe sunt:

$$x_P = x_1 + x_P^{'} = x_1 + \frac{\Delta y_{1-2} \cdot \left(D_{1-P}^2 + D_{1-3}^2 - D_{3-P}^2\right) - \Delta y_{1-3} \cdot \left(D_{1-P}^2 + D_{1-2}^2 - D_{2-P}^2\right)}{2 \cdot \left(\Delta x_{1-3} \cdot \Delta y_{1-2} - \Delta x_{1-2} \cdot \Delta y_{1-3}\right)}$$

(3.19b)

$$y_P = y_1 + y_P^{'} = y_1 + \frac{\Delta x_{1-3} \times \left(D_{1-P}^2 + D_{1-2}^2 - D_{2-P}^2\right) - \Delta x_{1-2} \times \left(D_{1-P}^2 + D_{1-3}^2 - D_{3-P}^2\right)}{2 \times \left(\Delta x_{1-3} \times \Delta y_{1-2} - \Delta x_{1-2} \times \Delta y_{1-3}\right)}$$

3.1.4 Calculul distanțelor şi orientărilor provizorii în rețea

După determinarea coordonatelor provizorii ale punctelor noi, pe baza acestora se calculează distanțele şi orientările provizorii, cu relațiile cunoscute (3.12 şi 3.16).

$$p_i = \frac{1}{(s_D^{'})^2} \tag{3.20}$$

unde $s_D^{'}$ este eroarea medie a seriei de observații efectuate asupra laturii respective.
În lipsa unor prelucrări prealabile ale observațiilor se poate estima o pondere a fiecărei distanțe măsurate de forma:

$$p_i = \frac{D_{\min}}{D_i} \tag{3.21}$$

unde D_{min} este lungimea celei mai mici laturi măsurate din rețea, care va primi ponderea 1.

3.3 Compensarea rețelelor de trilaterație prin metoda măsurătorilor indirecte ponderate

3.3.1 Aspecte ale ecuațiilor de corecții

În cazul metodei măsurătorilor indirecte , variația coordonatelor, fiecare măsurătoare de distanță generează o ecuație de corecție. **Considerând două puncte noi ale rețelei, *i* şi *k* , de coordonate provizorii x_i^0 , y_i^0 şi x_k^0 , y_k^0 şi distanța măsurată între aceste puncte, redusă la planul proiecției, D_{ik}^* , ecuația de corecție generată de această măsurătoare, pentru latura din rețea văzută de la punctul *i* către punctul *k* se scrie sub formă linearizată - prin introducerea coordonatelor provizorii ale punctelor noi- ca în relația de mai jos:**

$$D_{ik}^* + v_{ik} = \sqrt{\left[\left(x_k^0 + dx_k\right) - \left(x_i^0 + dx_i\right)\right]^2 + \left[\left(y_k^0 + dy_k\right) - \left(y_i^0 + dy_i\right)\right]^2} \tag{3.22}$$

După dezvoltarea relației (3.22) , aceasta devine:

$$\left(x_k^0 - x_i^0\right) \qquad \left(y_k^0 - y_i^0\right) \qquad \left(x_k^0 - x_i^0\right)$$

sau:

$$-\cos q_{ik}^{0} \times dx_i - \sin q_{ik}^{0} \times dy_i + \cos q_{ik}^{0} \times dx_k + \sin q_{ik}^{0} \times dy_k + l_{ik} = v_{ik} \qquad (3.24)$$

Dacă punctul *i* este nou şi punctul *k* este un punct vechi, atunci ecuația de corecție capătă forma:

$$-\cos q_{ik}^{0} \cdot dx_i - \sin q_{ik}^{0} \cdot dy_i + l_{ik} = v_{ik} \qquad (3.25)$$

Dacă punctul *i* este punct vechi şi punctul *k* este nou, atunci ecuația de corecție devine:

$$\cos q_{ik}^{0} \cdot dx_k + \sin q_{ik}^{0} \cdot dy_k + l_{ik} = v_{ik} \qquad (3.26)$$

Pentru cazul în care ambele puncte, *i* şi *k* sunt puncte vechi, nu se poate scrie o ecuație de corecție.

3.3.2 Calculul coeficienților ecuațiilor de corecții

Se observă că toți coeficienții ecuațiilor de corecții sunt valori ale funcțiilor *sin* şi *cos*, care pot varia în limitele 0…1. Acest lucru constituie o verificare a calculului acestora pentru crearea matricii ecuațiilor inițiale de corecții.

Din ecuațiile (3.23)-(3.26) se observă că valorile coeficienților pentru matricea ecuațiilor inițiale de corecții se pot calcula ca valori ale funcțiilor *sin* şi *cos* ale orientărilor provizorii, q_{ik}^{0} ale laturilor măsurate sau ca valori ale rapoartelor $A_{ik} = \frac{\Delta x_{ik}^{0}}{D_{ik}^{0}} = \cos q_{ik}^{0}$ şi $B_{ik} = \frac{\Delta y_{ik}^{0}}{D_{ik}^{0}} = \sin q_{ik}^{0}$, în care $\Delta x_{ik}^{0} = x_k^{0} - x_i^{0}$ şi $\Delta y_{ik}^{0} = y_k^{0} - y_i^{0}$.

De asemenea observă că termenii liberi ai ecuațiilor rezultă din diferența $l_{ik} = D_{ik}^{0} - D_{ik}^{*}$,în care D_{ik}^{0} este distanța provizorie (calculată cu coordonatele provizorii ale punctelor) şi D_{ik}^{*} este distanța măsurată pe teren, redusă la planul proiecției, pe latura respectivă.

$$-A_{ik}\cdot dx_i - B_{ik}\cdot dy_i + A_{ik}\cdot dx_k + B_{ik}\cdot dy_k + l_{ik} = v_{ik} \qquad (3.27)$$

- pentru punctul nou, *i* şi punctul vechi, *k*:

$$-A_{ik}\cdot dx_i - B_{ik}\cdot dy_i + l_{ik} = v_{ik} \qquad (3.28)$$

- pentru punctul vechi, *i* şi punctul nou, *k*:

$$A_{ik}\cdot dx_k + B_{ik}\cdot dy_k + l_{ik} = v_{ik} \qquad (3.29)$$

3.3.3 Formarea matricii ecuațiilor inițiale de corecții

Pentru o rețea de trilaterație cu *NPN* puncte noi, *NPV* puncte vechi şi *NLM* laturi măsurate, va rezulta un sistem de *NLM* ecuații liniare cu $2\cdot NPN$ necunoscute (corecțiile coordonatelor provizorii ale punctelor noi). În general, datorită surplusului de măsurători rezultă că $NLM \mathbf{f} 2\cdot NPN$, adică numărul de ecuații este mai mare decât numărul de necunoscute.

Fiecare ecuație va avea o pondere individuală, corespunzătoare măsurătorii respective de distanță, pondere calculată conform relațiilor (3.20) sau (3.21).

Asamblarea ecuațiilor de corecții pentru a crea matricea inițială se face în modul următor: se consideră un cap de tabel în care se trec toate punctele noi într-o ordine prestabilită; fiecărui punct îi vor corespunde două coloane(corecțiile necunoscute, *dx* şi *dy* ale coordonatelor provizorii ale punctului respectiv) , cum se vede în tabelul 3.1.

Aşa cum s-a arătat, orice latură măsurată, care are cel puțin un punct nou în unul dintre capetele sale generează o ecuație de corecții; să presupunem că prima latură măsurată are drept capete punctele noi cu numerele *1* şi *NPN*. Aceasta va genera ecuația nr. 1, cu coeficienții:

$$A_1 = \frac{\Delta x^0_{1-NPN}}{D^0_{1-NPN}} = \cos q^0_{1-NPN} \quad \text{şi} \quad B_1 = \frac{\Delta y^0_{1-NPN}}{D^0_{1-NPN}} = \sin q^0_{1-NPN} \qquad (3.30)$$

şi termenul liber:

2NPN-1 şi *2NPN*, corespunzătoare corecţiilor pentru punctul *NPN* se trec valorile $+A_1$, respectiv $+B_1$ şi în coloana *2NPN+1* se trece valoarea termenului liber, l_1. În celelalte coloane coeficienţii sunt zero.
Presupunem că ultima latură măsurată este cuprinsă între punctul nou *2* şi punctul vechi *4*. Aceasta generează ecuaţia numărul *NLM* cu coeficienţii:

$$A_{NLM} = \frac{\Delta x^0_{2-4}}{D^0_{2-4}} = \cos q^0_{2-4} \quad \text{şi} \quad B_{NLM} = \frac{\Delta y^0_{2-4}}{D^0_{2-4}} = \sin q^0_{2-4} \tag{3.32}$$

şi termenul liber:

$$l_{NLM} = D^0_{2-4} - D^*_{2-4} \tag{3.33}$$

În tabelul 3.1 se trec în linia *NLM* , în coloanele 3 şi 4(corespunzătoare punctului nou *2*) valorile $-A_{NLM}$, respectiv $-B_{NLM}$ şi, în coloana *2NPN+1* valoarea l_{NLM} , iar în celelalte coloane valoarea zero.
În mod asemănător se completează toate cele NLM linii ale matricii iniţiale ale ecuaţiilor de corecţii.

3.3.4 Formarea matricii ecuaţiilor transformate echivalente de corecţii

Aşa cum s-a arătat la cap. 3.2 , fiecare măsurătoare , deci fiecare ecuaţie de corecţii este caracterizată de o pondere individuală. Rezultă deci că ecuaţia nr. 1 are ponderea *p1*, ecuaţia nr. 2 are ponderea *p2* şi aşa mai departe. Pentru uniformizarea ponderilor tuturor ecuaţiilor, acestea se aduc la valoarea 1, prin aplicarea regulii a treia de echivalenţă a lui Schreiber, adică pentru fiecare ecuaţie se înmulţesc coeficienţii şi termenul liber cu radicalul ponderii individuale a ecuaţiei respective. În acest mod se obţine matricea ecuaţiilor transformate echivalente de corecţii, în care toate ecuaţiile au ponderea *p=1*, aşa cum se prezintă în tabelul 3.2:

Tabel 3.1 Matricea ecuațiilor inițiale

Număr ecuație	Punct nou 1		Punct nou 2		Punct nou 3		….		Punct nou NPN		Terme liber
	dx_1	dy_1	dx_2	dy_2	dx_3	dy_3			dx_{NPN}	dy_{NPN}	
0	1	2	3	4	5	6	..	..	2NPN-1	2NPN	2NPN+
ec. 1	$-A_1$	$-B_1$	0	0	0	0	0	0	A_1	B_1	l_1
ec. 2	..										
ec. 3	..										
ec. 4	..										
….	..										
….	..										
ec. NLM	0	0	$-A_{NLM}$	$-B_{NLM}$	0	0	0	0	0	0	l_{NLM}

Tabel 3.2 Matricea ecuațiilor transformate echivalente

Număr ecuație	Punct nou 1		Punct nou 2		….		Punct nou NPN		Te l
	dx_1	dy_1	dx_2	dy_2			dx_{NPN}	dy_{NPN}	
0	1	2	3	4			2NPN-1	2NPN	2N
ec. 1	$-A_1\sqrt{p_1}$	$-B_1\sqrt{p_1}$	0	0			$A_1\sqrt{p_1}$	$B_1\sqrt{p_1}$	l_1
ec. 2	..								
ec. 3	..								
….	..								
ec. NLM	$-A_{NLM}\sqrt{p_{NLM}}$	$-B_{NLM}\sqrt{p_{NLM}}$					0	0	l_{NLM}

3.3.5 Crearea sistemului liniar al ecuaţiilor normale ale corecţiilor coordonatelor punctelor noi

Sistemul ecuaţiilor iniţiale ale corecţiilor, de forma (3.27)-(3.29) , trecut în tabelul 3.1, se poate scrie şi astfel:

$$\begin{aligned}
b_{11}x_1 + b_{12}x_2 + b_{13}x_3 + b_{14}x_4 + \ldots\ldots\ldots + b_{1n}x_n + l_1 = v_1 \quad & \text{cu pondere } p_1 \\
b_{21}x_1 + b_{22}x_2 + b_{23}x_3 + b_{24}x_4 + \ldots\ldots\ldots + b_{2n}x_n + l_2 = v_2 \quad & \text{cu pondere } p_2 \\
\ldots\ldots\ldots

$$\mathbf{B}^T\mathbf{Pv}=0 \tag{3.38}$$

Din relaţiile (3.36) şi (3.38) rezultă sistemul ecuaţiilor normale ale corecţiilor coordonatelor provizorii ale punctelor noi:

$$\mathbf{B}^T\mathbf{PBx}+\mathbf{B}^T\mathbf{PL}=0 \tag{3.39}$$

Soluţia sistemului (3.39)-deci corecţiile coordonatelor provizorii ale punctelor noi –rezultă din:

$$\mathbf{x}=-(\mathbf{B}^T\mathbf{PB})^{-1}(\mathbf{B}^T\mathbf{PL})=-\mathbf{QB}^T\mathbf{PL} \tag{3.40}$$

unde $(\mathbf{B}^T\mathbf{PB})^{-1}=\mathbf{Q}$ este matricea inversă a sistemului ecuaţiilor normale, numită şi matricea coeficienţilor de pondere.
Corecţiile distanţelor măsurate se calculează conform relaţiei (3.36):

$$\mathbf{v}=\mathbf{Bx}+\mathbf{L}=-\mathbf{BQB}^T\mathbf{PL}+\mathbf{L}=(\mathbf{E}-\mathbf{BQB}^T\mathbf{P})\mathbf{L} \tag{3.41}$$

Pentru cazul observaţiilor considerate independente, sistemul (3.39) are ca echivalent:

$$\begin{aligned}
&\lfloor p_j \times b_{j1} \times b_{j1} \rfloor \times x_1 + \lfloor p_j \times b_{j1} \times b_{j2} \rfloor \times x_2 + .. + \lfloor p_j \times b_{j1} \times b_{jn} \rfloor \times x_n + \lfloor p_j \times b_{j1} \times l_j \rfloor = 0 \\
&\lfloor p_j \times b_{j2} \times b_{j1} \rfloor \times x_1 + \lfloor p_j \times b_{j2} \times b_{j2} \rfloor \times x_2 + .. + \lfloor p_j \times b_{j2} \times b_{jn} \rfloor \times x_n + \lfloor p_j \times b_{j2} \times l_j \rfloor = 0 \\
&\ldots\ldots\ldots

…….. (3.43)

$$\lfloor b_{jn} \cdot b_{j1} \rfloor \cdot x_1 + \lfloor b_{jn} \cdot b_{j2} \rfloor \cdot x_2 + \ldots\ldots\ldots + \lfloor b_{jn} \cdot b_{jn} \rfloor \cdot x_n + \lfloor b_{jn} \cdot l_j \rfloor = 0$$

unde toate ecuaţiile au ponderea unitară.

În ecuaţiile (3.42) şi (3.43) s-a folosit notaţia $[.....]$ în loc de $\sum_{j=1}^{m}(.....)$.

3.3.6 Calculul coordonatelor compensate ale punctelor noi

Soluţia sistemului (3.42) sau (3.43), $\mathbf{x}^T=[x_1, x_2, x_3, x_4, \ldots\ldots\ldots, x_{2NPN-1}, x_{2NPN}]$ reprezintă de fapt corecţiile coordonatelor provizorii ale punctelor noi, în ordinea în care ele au fost scrise în matricea ecuaţiilor iniţiale(v. tab. 3.1). Rezultă că soluţia respectivă este echivalentă cu $[\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2, \ldots, \Delta x_{NPN}, \Delta y_{NPN}]$. Prin urmare coordonatele compensate ale punctelor noi se vor calcula cu valorile de mai sus astfel:

$$\begin{aligned} X_1 &= x_1^0 + x_1 = x_1^0 + \Delta x_1 \\ Y_1 &= y_1^0 + x_2 = y_1^0 + \Delta y_1 \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ X_{NPN} &= x_{NPN}^0 + x_{2NPN-1} = x_{NPN}^0 + \Delta x_{NPN} \\ Y_{NPN} &= y_{NPN}^0 + x_{2NPN} = y_{NPN}^0 + \Delta y_{NPN} \end{aligned} \qquad (3.44)$$

3.3.7 Calculul distanţelor compensate

Corecţiile distanţelor se calculează cu relaţia (3.41) sau prin înlocuirea soluţiei sistemului normal în fiecare din ecuaţiile iniţiale. După calculul corecţiilor $v_1, v_2, \ldots v_m$, distanţele compensate se determină sub forma:

$$D_j = D_j^* + v_j \quad , \ j=1\ldots m \qquad (3.45)$$

Distanţele compensate trebuie să fie foarte puţin diferite faţă de cele

3.3.8 Evaluarea preciziei rezultatelor

Precizia globală a măsurătorilor din reţea se apreciază prin eroarea medie a unităţii de pondere:

$$s_0 = \sqrt{\frac{v^T \times P \times v}{m - n}} = \sqrt{\frac{[p \times v \times v]}{m - n}} \qquad (3.46)$$

Erorile de determinare a coordonatelor unui punct nou din reţea se calculează cu:

$$s_{x_k} = s_0 \cdot \sqrt{Q_{x_{2k-1}x_{2k-1}}} \quad \text{şi} \quad s_{y_k} = s_0 \cdot \sqrt{Q_{x_{2k}x_{2k}}} \qquad (3.47)$$

unde termenii $Q_{x_{2k-1}x_{2k-1}}$ şi $Q_{x_{2k}x_{2k}}$, k=1..n , se află pe diagonala principală a matricii coeficienţilor de pondere în poziţiile sugerate de index, ţinând cont de ordinea de scriere a punctelor noi în matricea ecuaţiilor iniţiale.

Precizia unei măsurători de distanţă se apreciază cu:

$$s_j = \frac{s_0}{\sqrt{p_j}} \text{ ,j=1…m} \qquad (3.48)$$

unde j este numărul distanţei măsurate(numărul ecuaţiei de corecţie) şi p_j-ponderea măsurătorii respective.

Cap. 4 Exemplu de calculul şi compensare a unei rețele de trilaterație

4.1 Datele de bază

În cele ce urmează se prezintă un exemplu de calcul referitor la îndesirea unei rețele planimetrice de sprijin prin lucrări de trilaterație.
Rețeaua se compune din 3 puncte vechi şi va fi completată cu un număr de 3 puncte noi, aşa cum se vede în fig. 4.1, care prezintă schița cu amplasamentul punctelor vechi şi noi şi distanțele măsurate.

Coordonatele punctelor vechi şi distanțele măsurate pe teren, reduse la planul proiecției sunt prezentate în tabelele 4.1 şi 4.2.

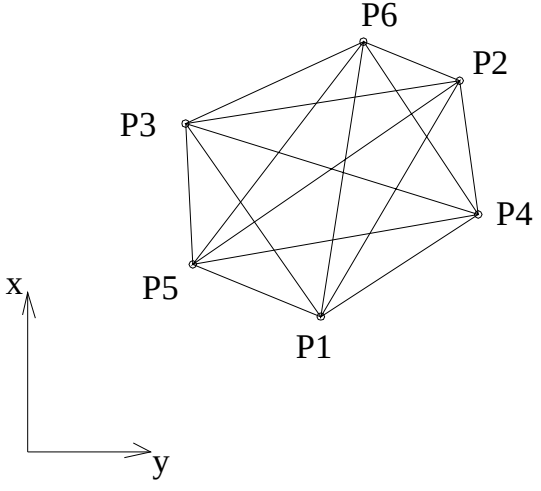

Fig. 4.1 Schița rețelei de trilaterație compensate

Tabel 4.1 Coordonatele punctelor de sprijin(vechi)

Nume Punct	Coordonate (m)	
	X	Y

Tabel 4.2 Distanţe măsurate reduse la planul proiecţiei

Nr. crt.	Latura măsurată		Distanţa măsurată (m)
	De la	Până la	
1	P1	P4	302.4192
2	P1	P5	177.8088
3	P1	P6	449.0329
4	P2	P4	179.2347
5	P2	P5	299.0678
6	P2	P6	169.4128
7	P3	P4	463.8445
8	P3	P5	173.7280
9	P3	P6	284.0398
10	P4	P1	302.4184
11	P4	P2	179.2355
12	P4	P3	463.8458
13	P5	P1	177.8085
14	P5	P2	299.0677
15	P5	P3	173.7299
16	P6	P1	449.0321
17	P6	P2	169.4136
18	P6	P3	284.0405
19	P4	P5	351.1839
20	P4	P6	348.4563
21	P5	P4	351.1831
22	P5	P6	331.2806
23	P6	P4	348.4555
24	P6	P5	331.2786

4.2 Calcule preliminarii

Cu datele prezentate la 4.1 s-au calculat -folosind una din posibilităţile oferite de relaţiile (3.12)-(3.19)- coordonatele provizorii şi distanţele şi orientările provizorii, care sunt trecute în tabelele 4.3 şi 4.4.

Tabel 4.3 Coordonatele provizorii ale punctelor noi

Nume Punct	Coordonate provizorii(m)	
	x^0	y^0

Tabel 4.4 Distanțe şi orientări provizorii

Nr. crt.	Latură rețea		Distanța provizorie (m)	Orientare provizorie (gon)
	De la	La		
1	P1	P4	302.4192	67.4673
2	P1	P5	177.8088	367.3155
3	P1	P6	449.0329	11.0267
4	P2	P4	179.2347	166.1251
5	P2	P5	299.0678	267.1721
6	P2	P6	169.4128	361.8920
7	P3	P4	463.8459	122.2195
8	P3	P5	173.7277	167.5511
9	P3	P6	284.0408	68.2691
10	P4	P1	302.4192	267.4673
11	P4	P2	179.2347	366.1251
12	P4	P3	463.8459	322.2195
13	P5	P1	177.8088	167.3155
14	P5	P2	299.0678	67.1721
15	P5	P3	173.7277	367.5511
16	P6	P1	449.0329	211.0267
17	P6	P2	169.4128	161.8920
18	P6	P3	284.0408	268.2691
19	P4	P5	351.1834	301.2656
20	P4	P6	348.4550	364.0682
21	P5	P4	351.1834	101.2656
22	P5	P6	331.2814	33.1287
23	P6	P4	348.4550	164.0682
24	P6	P5	331.2814	233.1287

4.3 Calculul coeficienților şi termenilor liberi ai ecuațiilor inițiale de corecții

Utilizând relațiile de tipul 3.30 şi 3.31, cu elementele de care se dispune până în această etapă s-au calculat coeficienții şi termenii liberi ai ecuațiilor inițiale de corecții. Numărul de ecuații este egal cu numărul laturilor măsurate deoarece toate acestea au cel puțin un punct nou în unul dintre capete. Coeficienții şi termenii liberi ai ecuațiilor inițiale sunt prezentați în tabelul 4.5.

4.5 Formarea matricii ecuațiilor inițiale de corecții şi a matricii ecuațiilor transformate echivalente

Asamblarea ecuațiilor de corecții în matricea ecuațiilor inițiale este prezentată în tabelul 4.6. Matricea ecuațiilor transformate echivalente se obține prin produsul termenilor de pe liniile matricii inițiale cu radicalul ponderii de pe linia respectivă. În acest fel ponderile devin 1.0 la toate ecuațiile şi coeficienții se modifică aşa cum se vede în tabelul 4.7.

Tabel 4.5 Coeficienții, termenii liberi şi ponderile ecuațiilor inițiale de corecții

Nr. ecuație	Latură rețea		Coeficient A=cos q_i^0	Coeficient B=sin q_i^0	Termen liber l= $D_i^0 - D_i^*$	Pondere $p_I=D_{min}/D_i$
	De la	la				
1	P1	P4	0.489069	0.872245	-0.0000	0.56019
2	P1	P5	0.871076	-0.491148	-0.0000	0.95278
3	P1	P6	0.985037	0.172342	0.0000	0.37728
4	P2	P4	-0.861741	0.507349	-0.0000	0.94520
5	P2	P5	-0.493108	-0.869968	0.0000	0.56647
6	P2	P6	0.826126	-0.563486	0.0000	1.00000
7	P3	P4	-0.341980	0.939707	0.0014	0.36524
8	P3	P5	-0.872888	0.487921	-0.0003	0.97516
9	P3	P6	0.478045	0.878335	0.0010	0.59644
10	P4	P1	-0.489069	-0.872245	0.0008	0.56019
11	P4	P2	0.861741	-0.507349	-0.0008	0.94520
12	P4	P3	0.341980	-0.939707	0.0001	0.36524
13	P5	P1	-0.871076	0.491148	0.0003	0.95278
14	P5	P2	0.493108	0.869968	0.0001	0.56647
15	P5	P3	0.872888	-0.487921	-0.0022	0.97515
16	P6	P1	-0.985037	-0.172342	0.0008	0.37728
17	P6	P2	-0.826126	0.563486	-0.0008	1.00000
18	P6	P3	-0.478045	-0.878335	0.0003	0.59644
19	P4	P5	0.019879	-0.999802	-0.0005	0.48240
20	P4	P6	0.844901	-0.534922	-0.0013	0.48618
21	P5	P4	-0.019879	0.999802	0.0003	0.48241
22	P5	P6	0.867628	0.497214	0.0008	0.51139

Tabel 4.6 Matricea ecuațiilor inițiale

Număr ecuație	Punct nou P4		Punct nou P5		Punct nou P6		Termen liber	Pondere
	dx_1	dy_1	dx_2	dy_2	dx_3	dy_3		
0	1	2	3	4	5	6	7	8
ec. 1	0.489069	0.872245	0	0	0	0	0	0.56019
ec. 2	0	0	0.871076	-0.491148	0	0	0	0.95278
ec. 3	0	0	0	0	0.985037	0.172342	0	0.37728
ec. 4	-0.861741	0.507349	0	0	0	0	0	0.94520
ec. 5	0	0	-0.493108	-0.869968	0	0	0	0.56647
ec. 6	0	0	0	0	0.826126	-0.563486	0	1.00000
ec. 7	-0.341980	0.939707	0	0	0	0	0.0014	0.36524
ec. 8	0	0	-0.872888	0.487921	0	0	-0.0003	0.97516
ec. 9	0	0	0	0	0.478045	0.878335	0.0010	0.59644
ec. 10	0.489069	0.872245	0	0	0	0	0.0008	0.56019
ec. 11	-0.861741	0.507349	0	0	0	0	-0.0008	0.94520
ec. 12	-0.341980	0.939707	0	0	0	0	0.0001	0.36524
ec. 13	0	0	0.871076	-0.491148	0	0	0.0003	0.95278
ec. 14	0	0	-0.493108	-0.869968	0	0	0.0001	0.56647
ec. 15	0	0	-0.872888	0.487921	0	0	-0.0022	0.97515
ec. 16	0	0	0	0	0.985087	0.172342	0.0008	0.37728
ec. 17	0	0	0	0	0.826126	-0.563486	-0.0008	1.00000
ec. 18	0	0	0	0	0.478045	0.878335	0.0003	0.59644
ec. 19	-0.019879	0.999802	0.019879	-0.999802	0	0	-0.0005	0.48240
ec. 20	-0.844901	0.534922	0	0	0.844901	-0.534922	-0.0013	0.48618
ec. 21	-0.019879	0.999802	0.019879	-0.999802	0	0	0.0003	0.48241
ec. 22	0	0	-0.867628	-0.497214	0.867628	0.497214	0.0008	0.51139
ec. 23	-0.844901	0.534922	0	0	0.844901	-0.534922	-0.0005	0.48618
ec. 24	0	0	-0.867628	-0.497214	0.867628	0.497214	0.0028	0.51139

Tabel 4.7 Matricea ecuațiilor transformate echivalente

Număr ecuație	Punct nou P4		Punct nou P5		Punct nou P6		Termen liber	Ponder
	dx_1	dy_1	dx_2	dy_2	dx_3	dy_3		
0	1	2	3	4	5	6	7	8
ec. 1	0.366049	0.652840	0	0	0	0	0	1.0000
ec. 2	0	0	0.850262	-0.479412	0	0	0	1.0000
ec. 3	0	0	0	0	0.605044	0.105858	0	1.0000
ec. 4	-0.837797	0.493252	0	0	0	0	0	1.0000
ec. 5	0	0	-0.371134	-0.654774	0	0	0	1.0000
ec. 6	0	0	0	0	0.826126	-0.563486	0	1.0000
ec. 7	-0.206675	0.567910	0	0	0	0	0.0008	1.0000
ec. 8	0	0	-0.861979	0.481823	0	0	-0.0003	1.0000
ec. 9	0	0	0	0	0.369192	0.678334	0.0008	1.0000
ec. 10	0.366049	0.652841	0	0	0	0	0.0006	1.0000
ec. 11	-0.837795	0.493251	0	0	0	0	-0.0008	1.0000
ec. 12	-0.206674	0.567909	0	0	0	0	0.0000	1.0000
ec. 13	0	0	0.850262	-0.479412	0	0	0.0003	1.0000
ec. 14	0	0	-0.371134	-0.654774	0	0	0.0001	1.0000
ec. 15	0	0	-0.861974	0.481821	0	0	-0.0022	1.0000
ec. 16	0	0	0	0	0.605044	0.105859	0.0005	1.0000
ec. 17	0	0	0	0	0.826124	-0.563484	-0.0008	1.0000
ec. 18	0	0	0	0	0.369192	0.678334	0.0003	1.0000
ec. 19	-0.013807	0.694416	0.013807	-0.694416	0	0	-0.0004	1.0000
ec. 20	-0.589122	0.372983	0	0	0.589122	-0.372983	-0.0009	1.0000
ec. 21	-0.013807	0.694417	0.013807	-0.694417	0	0	0.0002	1.0000
ec. 22	0	0	-0.620453	-0.355564	0.620453	0.355564	0.0006	1.0000
ec. 23	-0.589122	0.372984	0	0	0.589122	-0.372984	-0.0004	1.0000
ec. 24	0	0	-0.620455	-0.355566	0.620455	0.355566	0.0020	1.0000

4.6 Scrierea sistemului ecuaţiilor normale

În matricea ecuaţiilor transformate notăm coeficienţii din coloanele 1…6 cu b_{j1} , b_{j2} , b_{j3} ,b_{j4} , b_{j5} , b_{j6} şi termenii liberi din coloana 7 cu l_j , în care j=1…24..Aducerea acestui sistem de 24 ecuaţii liniare cu 6 necunoscute la forma normală (număr ecuaţii = număr necunoscute) adică:

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + ...+ a_{16}x_6 + L_1 = 0\\
a_{21}x_1 + a_{22}x_2 + ...+ a_{26}x_6 + L_2 = 0\\
a_{31}x_1 + a_{32}x_2 + ...+ a_{36}x_6 + L_3 = 0\\
a_{41}x_1 + a_{42}x_2 + ...+ a_{46}x_6 + L_4 = 0\\
a_{51}x_1 + a_{52}x_2 + ...+ a_{56}x_6 + L_5 = 0\\
a_{61}x_1 + a_{62}x_2 + ...+ a_{66}x_6 + L_6 = 0
\end{aligned}$$

presupune calculul coeficienţilor a_{ik} şi al termenilor liberi L_i , din sistemul de ecuaţii normale cu relaţiile :

$$a_{11} = \sum_{j=1}^{24}\left(b_{j1}\cdot b_{j1}\right) \quad ; \quad a_{12} = a_{21} = \sum_{j=1}^{24}\left(b_{j1}\cdot b_{j2}\right) \quad ; \quad a_{13} = a_{31} = \sum_{j=1}^{24}\left(b_{j1}\cdot b_{j3}\right) \quad ;\text{.....};$$

$$a_{16} = a_{61} = \sum_{j=1}^{24}\left(b_{j1}\cdot b_{j6}\right) \; ; \; L_1 = \sum_{j=1}^{24}\left(b_{j1}\cdot l_j\right);$$

$$a_{21} = a_{12} = \sum_{j=1}^{24}\left(b_{j2}\cdot b_{j1}\right) \quad ; \quad a_{22} = \sum_{j=1}^{24}\left(b_{j2}\cdot b_{j2}\right) \quad ; \quad a_{23} = a_{32} = \sum_{j=1}^{24}\left(b_{j2}\cdot b_{j3}\right) \quad ;\text{.....};$$

$$a_{26} = a_{62} = \sum_{j=1}^{24}\left(b_{j2}\cdot b_{j6}\right) \; ; L_2 = \sum_{j=1}^{24}\left(b_{j2}\cdot l_j\right);$$

.....

.....;

$$a_{61} = a_{16} = \sum_{j=1}^{24}\left(b_{j6}\cdot b_{j1}\right) \; ; \; a_{62} = a_{26} = \sum_{j=1}^{24}\left(b_{j6}\cdot b_{j2}\right) \; ; \; a_{63} = a_{36} = \sum_{j=1}^{24}\left(b_{j6}\cdot b_{j3}\right) \; ;\text{.....};$$

$$a_{66} = \sum_{j=1}^{24}\left(b_{j6}\cdot b_{j6}\right) \; ; L_6 = \sum_{j=1}^{24}\left(b_{j6}\cdot l_j\right);$$

După efectuarea calculelor conform relaţiilor precedente au rezultat coeficienţii şi termenii liberi ai sistemului ecuaţiilor normale prezentaţi în tabelul 4.8. Se observă că matricea coeficienţilor ecuaţiilor normale este

4.7 Calculul coeficienților şi termenilor liberi ai sistemului redus şi găsirea soluției

Sistemul de ecuații normale se poate reduce la un sistem cu matrice a coeficienților superior triunghiulară printr-un algoritm de factorizare de tip Cholesky sau Gauss. Sistemul redus prin algoritmul Gauss se prezintă în tabelul 4.9.

Rezolvarea sistemului s-a realizat prin inversarea matricii coeficienților. Matricea inversă (care este matricea coeficienților de pondere Q_{ik}) este prezentată în tabelul 4.10.

Tabel 4.8 Coeficienţii şi termenii liberi din sistemul ecuaţiilor normale

Nr. ec.	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	L_i
1	**2.451726**	-1.041931	-0.000381	0.019175	-0.694129	0.439466	**-0.0014**
2	-1.041931	**3.226702**	0.019175	-0.964429	0.439466	-0.278234	**0.00007**
3	-0.000381	0.019175	**3.977686**	-0.737824	-0.769925	-0.441223	**-0.0007**
4	0.019175	-0.964429	-0.737824	**2.998717**	-0.441223	-0.252853	**0.00222**
5	-0.694129	0.439466	-0.769925	-0.441223	**3.833780**	-0.300292	**-0.0009**
6	0.439466	-0.278234	-0.441223	-0.252853	-0.300292	**2.108803**	**-0.0026**

Tabel 4.9 Coeficienţii şi termenii liberi ai sistemului redus

Nr. ec.	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	L_i
1	**2.451726**	-1.041931	-0.000381	0.019175	-0.694129	0.439466	**-0.0014**
2	-	**2.783904**	0.019013	-0.956280	0.144476	-0.091470	**-0.0005**
3	-	-	**3.977556**	-0.731290	-0.771020	-0.440530	**-0.0007**
4	-	-	-	**2.535631**	-0.527922	-0.368704	**0.00191**
5	-	-	-	-	**3.370391**	-0.333282	**-0.0010**
6	-	-	-	-	-	**1.891665**	**-0.0022**

Tabel 4.10 Matricea inversă a sistemului redus

Q_{i1}	Q_{i2}	Q_{i3}	Q_{i4}	Q_{i5}	Q_{i6}
0.407876	0.152656	-0.000691	0.054288	0.085803	-0.061837
-	**0.359208**	-0.001717	0.134975	0.005351	0.044220
-	-	**0.251411**	0.072508	0.068871	0.084815
-	-	-	**0.394379**	0.061774	0.087752
-	-	-	-	**0.296701**	0.052274
-	-	-	-	-	**0.528635**

Soluţia sistemului se prezintă în tabelul 4.11. Valorile rezultate reprezintă corecţiile coordonatelor provizorii ale punctelor noi.

Tabel 4.11 Soluţia sistemului

$x_1=dx_{P4}$= - 0.0005
$x_2=dy_{P4}$= 0.0000
$x_3=dx_{P5}$= - 0.0003
$x_4=dy_{P5}$= 0.0005
$x_5=dx_{P6}$= - 0.0004
$x_6=dy_{P6}$= - 0.0012

4.8 Calculul coordonatelor compensate ale punctelor noi

Cu coordonatele provizorii din tabelul 4.3 şi soluţia sistemului din tabelul 4.11 se calculează coordonatele definitive ale punctelor noi, după relaţia (3.44), astfel:

$$X_{P4} = 8504.0970 - 0.0005 = 8504.0965$$
$$Y_{P4} = 8978.7146 + 0.0000 = 8978.7145$$
$$X_{P5} = 8511.0780 - 0.0003 = 8511.077$$
$$Y_{P5} = 8627.6006 + 0.0005 = 8627.6011$$
$$X_{P6} = 8798.5071 - 0.0004 = 8798.5067$$
$$Y_{P6} = 8792.3183 - 0.0012 = 8792.3171$$

4.9 Calculul corecţiilor distanţelor măsurate şi al distanţelor compensate

Prin înlocuirea soluţiei sistemului în ecuaţiile iniţiale se determină corecţiile distanţelor măsurate sub forma:

$$v_j = b_{j1} \times x_1 + b_{j2} \times x_2 + b_{j3} \times x_3 + b_{j4} \times x_4 + b_{j5} \times x_5 + b_{j6} \times x_6 + l_j$$

Distanţele compensate se calculează cu relaţia generală:

$$D_{ij\ comp} = D_{ij}^{*} + v_{ij}$$

Tabel 4.12 Corecţiile distanţelor măsurate , distanţele compensate şi factorii pvv

Nr. crt.	Latura măsurată		Dist. măsurată D_j^* (m)	Pondere p_i	Corecție v_i	Dist. compens. (m)	$p_j v_j v_j$
1	P1	P4	302.4192	0.56019	-0.0003	302.4189	$5.0417\ 10^{-8}$
2	P1	P5	177.8088	0.95278	-0.0005	177.8083	$23.8195\ 10^{-8}$
3	P1	P6	449.0329	0.37728	-0.0006	449.0323	$13.5821\ 10^{-8}$
4	P2	P4	179.2347	0.94520	0.0004	179.2351	$15.1232\ 10^{-8}$
5	P2	P5	299.0678	0.56647	-0.0003	299.0675	$5.0982\ 10^{-8}$
6	P2	P6	169.4128	1.00000	0.0003	169.4131	$9.0000\ 10^{-8}$
7	P3	P4	463.8445	0.36524	0.0015	463.8460	$82.1790\ 10^{-8}$
8	P3	P5	173.7280	0.97516	0.0002	173.7282	$3.9006\ 10^{-8}$
9	P3	P6	284.0398	0.59644	-0.0002	284.0396	$2.3858\ 10^{-8}$
10	P4	P1	302.4184	0.56019	0.0005	302.4189	$14.0048\ 10^{-8}$
11	P4	P2	179.2355	0.94520	-0.0004	179.2351	$15.1232\ 10^{-8}$
12	P4	P3	463.8458	0.36524	0.0002	463.8460	$1.4610\ 10^{-8}$
13	P5	P1	177.8085	0.95278	-0.0002	177.8083	$3.8111\ 10^{-8}$
14	P5	P2	299.0677	0.56647	-0.0002	299.0675	$2.2659\ 10^{-8}$
15	P5	P3	173.7299	0.97515	-0.0017	173.7282	$281.8184\ 10^{-8}$
16	P6	P1	449.0321	0.37728	0.0002	449.0323	$1.5091\ 10^{-8}$
17	P6	P2	169.4136	1.00000	-0.0005	169.4131	$25.0000\ 10^{-8}$
18	P6	P3	284.0405	0.59644	-0.0009	284.0396	$48.3116\ 10^{-8}$
19	P4	P5	351.1839	0.48240	-0.0010	351.1829	$48.2400\ 10^{-8}$
20	P4	P6	348.4563	0.48618	-0.0006	348.4557	$17.5025\ 10^{-8}$
21	P5	P4	351.1831	0.48241	-0.0002	351.1829	$1.9296\ 10^{-8}$
22	P5	P6	331.2806	0.51139	-0.0001	331.2805	$0.5114\ 10^{-8}$
23	P6	P4	348.4555	0.48618	0.0002	348.4557	$1.9447\ 10^{-8}$
24	P6	P5	331.2786	0.51139	0.0019	331.2805	$184.6118\ 10^{-8}$

$\sum = \mathbf{808.1752\ 10^{-8}}$

Distanțele compensate se compară cu distanțele calculate din coordonatele definitive. Între valorile corespondente nu trebuie să fie diferențe mai mari de 10^{-5} m , acest fapt fiind o verificare a corectitudinii calculului de compensare.

4.10 Evaluarea preciziei rezultatelor

Eroarea medie a unității de pondere, adică precizia globală a măsurătorilor din rețea, a rezultat din relația:

$$s_0 = \sqrt{\frac{\mathring{a}\ pvv}{n - u}} = \sqrt{\frac{808.1752 \times 10^{-8}}{24 - 6}} = 0.00067 \text{ m}$$

Eroarea minimă şi eroarea maximă de măsurare a unei distanțe s-au calculat cu:

$$S_{D\max} = \frac{s_0}{\sqrt{p_{\min}}} = \frac{0.00067}{\sqrt{0.3652}} = 0.0011 \text{ m}$$

$$S_{D\min} = \frac{s_0}{\sqrt{p_{\max}}} = \frac{0.00067}{\sqrt{1}} = 0.00067 \text{ m}$$

Erorile de determinare a punctelor noi s-au calculat cu relațiile:

$$S_{N_{P4}} = s_0 \times \sqrt{Q_{11}} = 0.00067 \times \sqrt{0.407876} = 0.000431$$

$$S_{E_{P4}} = s_0 \times \sqrt{Q_{22}} = 0.00067 \times \sqrt{0.359208} = 0.000404$$

$$S_{N_{P5}} = s_0 \times \sqrt{Q_{33}} = 0.00067 \times \sqrt{0.251411} = 0.000338$$

$$S_{E_{P5}} = s_0 \times \sqrt{Q_{44}} = 0.00067 \times \sqrt{0.394379} = 0.000423$$

$$S_{N_{P6}} = s_0 \times \sqrt{Q_{55}} = 0.00067 \times \sqrt{0.296701} = 0.000367$$

$$S_{E_{P6}} = s_0 \times \sqrt{Q_{66}} = 0.00067 \times \sqrt{0.528635} = 0.000490$$

Cap. 5 Schemă logică generală şi program în limbaj Borland Pascal 7.0, pentru calculul şi compensarea reţelelor de trilateraţie

5.1 Prezentare

În cele ce urmează se prezintă automatizarea calculului şi compensării unei reţele de trilateraţie oarecare, prin intermediul unui program pentru calculator. Codul sursă este scris în limbajul de programare Pascal 7.0 pentru Windows, la nivelul anului 1999 şi compilarea este realizată în modul real, deci executabilul rezultat poate lucra în mediul de operare DOS şi sub Windows98. Programul poate fi însă uşor adaptat la posibilităţile grafice actuale ale tehnicii de calcul prin rescriere în limbaj C++.

Programul permite introducerea datelor iniţiale de la tastatură sau din fişiere de date mai vechi, create prin intermediul acestuia, afişarea şi corecţia datelor, calculul şi compensarea reţelei, afişarea rezultatelor, salvarea în fişiere şi tipărirea lor. Pentru scopuri didactice, dar şi de cercetare, salvarea, afişarea şi tipărirea rezultatelor se referă şi la etapele intermediare ale calculului, fapt prin care programul conferă posibilităţi de analiză în anumite situaţii.

Pentru asigurarea unei precizii de calcul sporite unele secţiuni, ca de exemplu cea referitoare la rezolvarea sistemelor de ecuaţii liniare folosesc opţiunea de lucru cu variabile în dublă precizie.

Datele introduse de la tastatură se salvează automat în două fisiere binare cu acces direct, care pot fi redenumite, fără să se schimbe extensia acestora. Datele şi rezultatele se pot salva opţional şi sub formă tabelară, într-un fişier text, care poate fi afişat apoi cu un utilitar de redactare ca de exemplu Notepad.

Schema logică generală a programului se prezintă în fig. 5.1 şi textul sursă în paragraful 5.2.

În final, la paragraful 5.3 se prezintă tabelele cu *datele* şi *rezultatele* dintr-un exemplu de calcul al reţelei de trilateraţie prezentate în fig. 5.2, aşa cum sunt acestea scoase de program. Reţeaua din fig. 5.2 este formată din 6 puncte vechi (HUN, D1A, MCB, SAL, COT, T1P) şi 3 puncte noi (GRN,

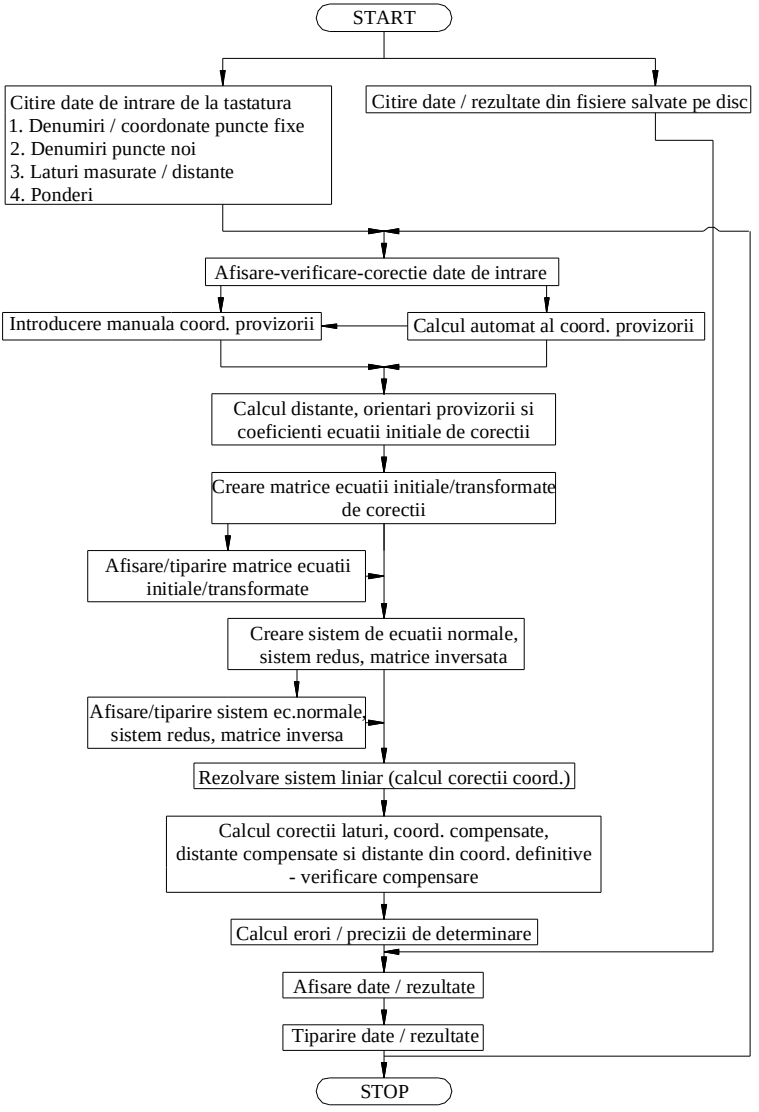
START
Citire date de intrare de la tastatura
1. Denumiri / coordonate puncte fixe
2. Denumiri puncte noi
3. Laturi masurate / distante
4. Ponderi
Citire date / rezultate din fisiere salvate pe disc
Afisare-verificare-corectie date de intrare
Introducere manuala coord. provizorii
Calcul automat al coord. provizorii
Calcul distante, orientari provizorii si coeficienti ecuatii initiale de corectii
Creare matrice ecuatii initiale/transformate de corectii
Afisare/tiparire matrice ecuatii initiale/transformate
Creare sistem de ecuatii normale, sistem redus, matrice inversata
Afisare/tiparire sistem ec.normale, sistem redus, matrice inversa
Rezolvare sistem liniar (calcul corectii coord.)
Calcul corectii laturi, coord. compensate, distante compensate si distante din coord. definitive - verificare compensare
Calcul erori / precizii de determinare
Afisare date / rezultate
Tiparire date / rezultate
STOP

Fig. 5.1 Schema logică generală a programului de calcul şi compensare a retelelor de trilateratie

5.2 Textul programului sursă

```
PROGRAM TRILATERATIE;
{$N+}
USES
DOS,MYCRT,GRAPH,BGIDRV,BGIFNT1,BGIFNT2,PRINTER,FUNC,MES,RSEND
,TTRIL;

VAR I,J,K,I1,J1,K1,NRP,N:LONGINT;
    POZY,NPFC:INTEGER;
    DX,DY,X1,X2,Y1,Y2,XP,YP,DB,D1,D2,D3:REAL;
    VV,EMP_UP:REAL;
    EI,ET:FISR;
    PCTN:STRING[6];PAR:STRING;

PROCEDURE PGSTART;
VAR
 GD,GM,CODCAR:INTEGER;
 ARC:REAL;
 REP,CUL,PAS,POL,X1,Y1,X2,Y2:INTEGER;
 NMDRV,NMFNT:STRING;
BEGIN
 GD:=REGISTERBGIDRIVER(@EGAVGADRV);GM:=GETGRAPHMODE;
 DETECTGRAPH(GD,GM);NMDRV:=GETDRIVERNAME;
 INITGRAPH(GD,GM,'NMDRV');
 IF GRAPHRESULT = GROK THEN BEGIN
  CUL:=7;PAS:=180;POL:=15;CLRSCR;
  SETGRAPHMODE(GETGRAPHMODE);
  SETBKCOLOR(LIGHTGREEN);CLEARVIEWPORT;SETCOLOR(WHITE);
  LINE(1,1,GETMAXX,1);LINE(GETMAXX,1,GETMAXX,GETMAXY);
  Line(1,GETMAXY,GETMAXX,GETMAXY);LINE(1,1,1,GETMAXY);
  X1 := 10;Y1 := 10;X2 := 629;Y2 := 469;
  SETCOLOR(BLUE);SETLINESTYLE(DottedLn, 0, NormWidth);
  RECTANGLE(X1, Y1, X2, Y2);
  SETLINESTYLE(UserBitLn, $C3, ThickWidth);
  RECTANGLE(Pred(X1), Pred(Y1), Succ(X2), Succ(Y2));
  SETLINESTYLE(0,0,1);SETCOLOR(CUL);
  REP:=0;ARC:=0;
  WHILE ARC<=2*PI DO BEGIN
   INC(REP);
   IF POL*REP>2*PAS THEN BEGIN
    INC(CUL);REP:=0;SETCOLOR(CUL);
   END;
```

```
 OUTTEXTXY(GETMAXX-210,GETMAXY-20,'ENTER PENTRU
 CONTINUARE !');
 SETTEXTSTYLE(0,0,1);SETCOLOR(RED);
 BAR(190,245,249,254);OUTTEXTXY(191,246,'1 9 9 9');
 CODCAR:=REGISTERBGIFONT(@GOTHFNT);
 SETTEXTSTYLE(CODCAR,0, 5);SETCOLOR(RED);
 OUTTEXTXY(250,20,'Calculul retelelor de');
 OUTTEXTXY(350,60,'Trilateratie');
 CODCAR:=REGISTERBGIFONT(@TRIPFNT);
 SETTEXTSTYLE(CODCAR,0,1);SETCOLOR(BLUE);
 OUTTEXTXY(373,120 ,'Realizat de :');
 OUTTEXTXY(384,140 ,'Conf.Dr.Ing. Adrian   Popia');
 SETCOLOR(YELLOW);
 OUTTEXTXY(392,160 ,'Universitatea Tehnica Iasi');
 OUTTEXTXY(420,180,'Catedra de Cadastru');
 READLN;CLOSEGRAPH;
 END ELSE WRITELN ('EROARE DRIVER GRAFIC !');
END;

PROCEDURE CAUT_FIS;
VAR I1:STRING[2];
BEGIN
 GETDIR(0,CALE);FSPLIT(CALE,NUMFIS,EXT);I:=9;
 REPEAT
  I:=I+1;STR(I:2,I1);NUMFIS:=CONCAT('TRILAT',I1);
  FIS1:=CONCAT(CALE,NUMFIS,'.PCT');
  FIS2:=CONCAT(CALE,NUMFIS,'.DIS');
  FIS:=FSEARCH(FIS1,CALE);
 UNTIL FIS='';
END;

FUNCTION CP(NMP:STRING):LONGINT;
VAR IND:LONGINT;
BEGIN
 CP:=0;
 FOR IND:=1 TO NPT DO BEGIN
  SEEK (C,IND);READ(C,LC);
  IF (LC.CODP=NMP) THEN CP:=IND;
 END;
END;
PROCEDURE CT1;
BEGIN
 REPEAT
```

```
 UNTIL Q='N';
END;
PROCEDURE C_T_1;
BEGIN
 WRITELN;TEXTCOLOR(RED); WRITELN ('INTRODUCERE DATE DE LA
 TASTATURA');
 WRITELN;SUN(1000,300);CAUT_FIS;
 ASSIGN(C,FIS1);ASSIGN(D,FIS2);REWRITE(C); REWRITE (D);
 WRITELN ('DATE GENERALE:'); WRITELN;TEXTCOLOR(BLUE);
 WRITE ('NUMARUL TOTAL DE PUNCTE VECHI ESTE:');
 SUN(500,300);READLN(NPV);LC.PV:=NPV;
 WRITE ('NUMARUL TOTAL DE PUNCTE NOI ESTE:');
 SUN(500,300);READLN(NPN);LC.PN:=NPN;
 NPT:=NPN+NPV;
 WRITELN;TEXTCOLOR(RED);
 WRITELN ('NUMELE PUNCTELOR VECHI :');
 WRITELN;TEXTCOLOR(BLUE);
 FOR I:=1 TO NPV DO BEGIN
  WRITE ('PUNCT VECHI ',I,' (max. 6 caract.):');SUN(600,300);
  READLN(LC.CODP);LC.X:=0;LC.Y:=0;LC.CX:=0;LC.CY:=0;
  SEEK (C,I); WRITE (C,LC);
 END;
 WRITELN;TEXTCOLOR(RED); WRITELN ('NUMELE PUNCTELOR NOI:');
 WRITELN;TEXTCOLOR(BLUE);
 FOR I:=1 TO NPN DO BEGIN
  WRITE ('PUNCT NOU ',I,' (max. 6 caract.) :');SUN(600,300);
  READLN(LC.CODP);LC.X:=0;LC.Y:=0; SEEK (C,NPV+I);
  WRITE (C,LC);
 END;
 CLRSCR;TEXTCOLOR(RED);CAP_T1; WRITELN (S0,'COORDONATE PUNCTE
 VECHI');
 TEXTCOLOR(YELLOW);SCR_CAP_T1; WRITELN (S0,S7);
 WRITELN (S0,S8);
 POZY:=WHEREY-2;TEXTCOLOR(BLUE);
 FOR I:=1 TO NPV DO BEGIN
  SEEK (C,I); READ (C,LC);GOTOXY(26,POZY); WRITELN (LC.CODP);
  CT1;
  SEEK (C,I); WRITE (C,LC);
 END;
END;

PROCEDURE CT2;
VAR I1:INTEGER;
```

```
  WRITE ('      ');GOTOXY(43,POZY); WRITE ('          ');
  REPEAT
   GOTOXY(27,POZY); WRITE ('      ');
   GOTOXY(27,POZY);READLN(LD.CODS);LD.TP1:=0;
   LD.TP1:=CP(LD.CODS); IF LD.TP1=0 THEN MESJ(27,10,MSG);
  UNTIL LD.TP1>0;
  REPEAT
   GOTOXY(35,POZY); WRITE ('      ');
   GOTOXY(35,POZY);READLN(LD.CODV);LD.TP2:=0;
   LD.TP2:=CP(LD.CODV); IF LD.TP2=0 THEN MESJ(27,10,MSG);
  UNTIL LD.TP2>0;
  GOTOXY(43,POZY);READLN(LD.DM);MESJR(60,10,MSG3);
 UNTIL Q='N';
END;

PROCEDURE C_T_2;
BEGIN
 CLRSCR;TEXTCOLOR(RED); WRITELN (S0,'INTRODUCERE DATE
 MASURATORI');
 TEXTCOLOR(YELLOW);CAP_T2;SCR_CAP_T2; WRITELN (S0,S7);
 WRITELN(S0,S8);
 I:=0;POZY:=WHEREY-2;TEXTCOLOR(BLUE);
 REPEAT
  I:=I+1;CT2; SEEK (D,I); WRITE (D,LD);
  MESJR(40,10,'SE INTRODUCE O NOUA LATURA ? D/N');
 UNTIL Q='N';
END;

PROCEDURE IN_P;
VAR PON,C1,C2,Dmin:REAL;
BEGIN
 WRITELN;TEXTCOLOR(RED);GOTOXY(25,WHEREY);
 WRITELN ('INTRODUCERE PONDERI MASURATORI');TEXTCOLOR(BLUE);
 GOTOXY(24,WHEREY); WRITELN ('1. MASURATORI DE ACEEASI
 PONDERE');
 GOTOXY(24,WHEREY); WRITELN ('2. MASURATORI DE PONDERI
 DIFERITE');
 GOTOXY(24,WHEREY); WRITELN ('   INTRODUSE INDIVIDUAL');
 GOTOXY(24,WHEREY); WRITELN ('3. PONDERI DIFERITE CALC.
 AUTOMAT CU');
 GOTOXY(24,WHEREY); WRITELN ('   RELATIA
 Pi=1/((C1+C2*Di)*(C1+C2*Di))');
 GOTOXY(24,WHEREY); WRITELN ('4. PONDERI DIFERITE CALC.
```

```
 CASE ORD(Q) OF
  49:BEGIN
      WRITELN;SUN(1000,200);
      WRITE ('VALOAREA UNICA A PONDERII ESTE:');READLN(PON);
      FOR I:=1 TO NTL DO BEGIN
       SEEK (D,I); READ (D,LD);LD.P0:=PON; SEEK (D,I); WRITE
       (D,LD);
      END;
     END;
  50:BEGIN
      FOR I:=1 TO NTL DO BEGIN
       SEEK (D,I); READ (D,LD);
       WRITE ('LATURA ',LD.CODS,'-',LD.CODV,' ARE ');
       WRITE ('PONDEREA:');READLN(PON);LD.P0:=PON;
       SEEK (D,I); WRITE (D,LD);
      END;
     END;
  51:BEGIN
      WRITELN ('SE INTRODUC CONSTANTELE C1 SI C2 DIN RELATIA
      ARATATA:');
      WRITE ('C1=');READLN(C1); WRITE ('C2=');READLN(C2);
      FOR I:=1 TO NTL DO BEGIN
       SEEK (D,I); READ (D,LD);
       PON:=C1+C2*LD.DM;LD.P0:=1/PON/PON;
       SEEK(D,I); WRITE (D,LD);
      END;
     END;
  52:BEGIN
      SEEK(D,1); READ (D,LD);Dmin:=LD.DM;
      FOR I:=1 TO NTL DO BEGIN
       SEEK(D,I); READ (D,LD); IF LD.DM<Dmin THEN
       Dmin:=LD.DM;
      END;
      FOR I:=1 TO NTL DO BEGIN
       SEEK(D,I); READ (D,LD);LD.P0:=Dmin/LD.DM;SEEK(D,I);
       WRITE (D,LD);
      END;
     END;
 END;
END;

PROCEDURE C_D;
VAR XX:SEARCHREC;
```

```
  TEXTCOLOR(BLUE);
  WRITE ('NUME_FISIER: ');SUN(1000,300);READLN(FIS);
  FSPLIT(FIS,CALE,NUMFIS,EXT); IF CALE='' THEN CALE:=CALE1;
  FIS1:=CALE+'\'+NUMFIS+'.PCT';FIS2:=CALE+'\'+NUMFIS+'.DIS';
  FINDFIRST(FIS1,ANYFILE,XX);
  IF DOSERROR<>0 THEN BEGIN
   WRITELN ('FISIERUL ',FIS1,' NU A FOST GASIT !');
   SUN(1000,400);
  END;
  FINDFIRST(FIS2,$3F,XX);
  IF DOSERROR<>0 THEN BEGIN
   WRITELN ('FISIERUL ',FIS2,' NU A FOST GASIT !');
   SUN(1000,400);
  END;
 UNTIL DOSERROR=0;
 ASSIGN(C,FIS1);ASSIGN(D,FIS2);RESET(C);RESET(D); SEEK (C,1);
 READ(C,LC);
 NPV:=LC.PV;NPN:=LC.PN;NTL:=FILESIZE(D)-1;NPT:=NPV+NPN;
 WRITELN;
 IF NPT>0 THEN BEGIN
  WRITELN ('INREGISTRAREA ',NUMFIS,' CONTINE:');
  WRITELN ('       - ',NPT,' PUNCTE IN TOTAL;');
  WRITELN ('       - ',LC.PV,' PUNCTE VECHI;');
  WRITELN ('       - ',LC.PN,' PUNCTE NOI;');
  WRITELN ('       - ',NTL,' LATURI MASURATE');
 END ELSE BEGIN
  WRITELN; WRITELN ('FISIERUL ',NUMFIS,' NU ARE DATE !! ');
  MESJ(60,WHEREY,MSG1);HALT;
 END;
 N:=2*NPN;MESJ(60,WHEREY,MSG1);
END;

PROCEDURE S_D;
VAR XX:SEARCHREC;
    CALE1:DIRSTR;
BEGIN
 WRITELN ('DATELE SE GASESC IN FISIERELE:');
 WRITELN ('  -COORDONATE       :',FIS1);
 WRITELN ('  -LATURI MASURATE  :',FIS2);
 MESJR(40,WHEREY,'SCHIMBI NUMELE FISIERELOR ?? D/N');
 IF Q='D' THEN BEGIN
  WRITELN; WRITE ('TASTEAZA NUME_FISIER :');
  READLN(FIS);CLOSE(C);CLOSE(D);CALE1:=CALE;
```

```
  WRITELN ('  -COORDONATE       :',FIS1);
  WRITELN ('  -LATURI MASURATE  :',FIS2);
  MESJ(60,WHEREY+1,MSG1);
  ASSIGN(C,FIS1);ASSIGN(D,FIS2);RESET(C);RESET(D);
 END;
END;

PROCEDURE V_D;
VAR MSGA,MSGB,MSGC:STRING;ALT,ALT1:REAL;Q1:CHAR;
BEGIN
 MSGA:='VERIFICARE COORDONATE PUNCTE VECHI';
 MSGB:='VERIFICARE LATURI MASURATE';
 MSGC:='VERIFICARE ALTITUDINI';
 REPEAT
   CLRSCR;GOTOXY(35,3); WRITELN ('VERIFICARI');
   GOTOXY(22,5); WRITELN ('1. ',MSGA);
   GOTOXY(22,6); WRITELN ('2. ',MSGB);
   GOTOXY(22,7); WRITELN ('3. ',MSGC);
   GOTOXY(22,8); WRITELN ('4. NU SE MAI FAC MODIFICARI');
   WRITE ('ALEGE 1...4 :');READLN(Q1); WRITELN (Q1);
   IF Q1='1' THEN BEGIN
     CLRSCR;CAP_T1;TEXTCOLOR(RED);
     WRITELN (S0,MSGA);TEXTCOLOR(YELLOW);SCR_CAP_T1;
     WRITELN (S0,S7); WRITELN (S0,S8);POZY:=WHEREY-2;
     FOR I:=1 TO NPV DO BEGIN
       SEEK (C,I); READ (C,LC);
       GOTOXY(26,POZY); WRITELN (LC.CODP:6);
       GOTOXY(34,POZY);WRITE('          ');
       GOTOXY(34,POZY); WRITELN (LC.X:10:3);
       GOTOXY(46,POZY);WRITE('          ');
       GOTOXY(46,POZY); WRITELN (LC.Y:10:3);
       MESJR(60,POZY+3,MSG3);
       IF Q='D' THEN BEGIN CT1; SEEK (C,I); WRITE (C,LC);END;
     END;
   END;
   IF Q1='2' THEN BEGIN
     CLRSCR;TEXTCOLOR(RED);CAP_T2;
     WRITELN (S0,MSGB);
     TEXTCOLOR(YELLOW);SCR_CAP_T2;
     WRITELN (S0,S7); WRITELN (S0,S8);POZY:=WHEREY-2;
     FOR I:=1 TO NTL DO BEGIN
       SEEK (D,I); READ (D,LD);GOTOXY(27,POZY);
       WRITELN ('      ');
```

```
        GOTOXY(35,POZY); WRITELN (LD.CODV:6);
        GOTOXY(43,POZY); WRITELN (LD.DM:10:4);
        MESJR(60,POZY+7,MSG3);
        IF Q='D' THEN BEGIN CT2; SEEK (D,I); WRITE (D,LD);END;
      END;
    END;
    IF Q1='3' THEN BEGIN
      ALT:=0;ALT1:=0; SEEK (D,0); READ (D,LD);ALT:=LD.TP3;
      IF ALT=0 THEN BEGIN
        SEEK (D,1); READ (D,LD);ALT1:=LD.TP3;
      END;
      IF (ALT<>0) OR (ALT1<>0) THEN BEGIN
        CLRSCR;TEXTCOLOR(RED);
        WRITELN (S0,MSGC);TEXTCOLOR(YELLOW);
        IF ALT<>0 THEN BEGIN
          WRITE ('ALTITUDINEA MEDIE A ZONEI ESTE:',ALT:5,
          ' SE MODIFICA ?? D/N');RASP;
          IF Q='D' THEN BEGIN
            SEEK (D,0); READ (D,LD);
            WRITE ('ALTIT. ELIPS. MEDIE A ZONEI DE MASURATORI
            ESTE:');
            READLN(ALT);LD.TP3:=ROUND(ALT); SEEK (D,0); WRITE
            (D,LD);
          END;
        END;
      END;
      IF ALT1<>0 THEN FOR I:=1 TO NTL DO BEGIN
        SEEK (D,I); READ (D,LD);
        WRITE ('ALTIT. LATURII ',LD.CODS,'-',LD.CODV,' ESTE:',
        LD.TP3:5,
        ' SE MODIFICA ?? D/N');RASP;
        IF Q='D' THEN BEGIN
          WRITE ('LATURA ',LD.CODS,'-',LD.CODV);
          WRITE ('  ALTIT. ELIPS. MEDIE=');
          READLN(ALT);LD.TP3:=ROUND(ALT);
          SEEK (D,I); WRITE (D,LD);
        END;
      END;
    END;
  UNTIL (Q1='4');
END;

PROCEDURE P_N_F_C;
```

```
 END;
END;

PROCEDURE IN_C_PR(II:LONGINT);
BEGIN
  SEEK (C,II); READ (C,LC);
  WRITE ('SE INTRODUC COORDONATE PROVIZORII PENTRU PUNCTUL');
  TEXTCOLOR(RED); WRITELN (LC.CODP,':');
  TEXTCOLOR(BLUE);POZY:=WHEREY;
  WRITE ('  X=');READLN(LC.X);GOTOXY(20,POZY);
  WRITE ('  Y=');READLN(LC.Y);
  WRITELN; SEEK (C,II); WRITE (C,LC);
END;

{PROCEDURE INT_DIR_DIS;
VAR NRP1,NRP2,NRP3:LONGINT;
    D12P,D13P,DX12,DY12,DX13,DY13,PA1,PA2,AUX,X3,Y3:REAL;
    PCT1,PCT2,PCT3:STRING[6];
BEGIN
 NRP1:=0;NRP2:=0;NRP3:=0;XP:=0;YP:=0;J:=0;I1:=0;
 X1:=0;Y1:=0;X2:=0;Y2:=0;X3:=0;Y3:=0;
 REPEAT
  J:=J+1;SEEK(D,J);READ(D,LD);
  IF (LD.TP1=NRP) OR (LD.TP2=NRP) THEN BEGIN
  IF NRP1=0 THEN BEGIN
    IF LD.TP1=NRP THEN NRP1:=LD.TP2;
    IF LD.TP2=NRP THEN NRP1:=LD.TP1;
    SEEK(C,NRP1);READ(C,LC);
    IF (LC.X<>0) THEN BEGIN
     PCT1:=LC.CODP;X1:=LC.X;Y1:=LC.Y;D1:=LD.DMR;I1:=I1+1;
    END ELSE BEGIN NRP1:=0;D1:=0;END;
  END;
  IF (NRP2=0) THEN BEGIN
    IF LD.TP1=NRP THEN NRP2:=LD.TP2;
    IF LD.TP2=NRP THEN NRP2:=LD.TP1;
    IF NRP2<>NRP1 THEN BEGIN
     SEEK(C,NRP2);READ(C,LC);
     IF (LC.X<>0) THEN BEGIN
      X2:=LC.X;Y2:=LC.Y;D2:=LD.DMR;PCT2:=LC.CODP;I1:=I1+1;
     END ELSE BEGIN NRP2:=0;D2:=0;END;
    END ELSE NRP2:=0;
  END;
  IF (NRP3=0) THEN BEGIN
```

```
      X3:=LC.X;Y3:=LC.Y;D3:=LD.DMR;PCT3:=LC.CODP;I1:=I1+1;
     END ELSE BEGIN NRP3:=0;D3:=0;END;
    END ELSE NRP3:=0;
  END;
  END;
 UNTIL (J=NTL) OR (I1=3);
 IF (I1=3) THEN BEGIN
  DX12:=X2-X1;DY12:=Y2-Y1;DX13:=X3-X1;DY13:=Y3-Y1;
  AUX:=2*(DX13*DY12-DX12*DY13);
  D12P:=DX12*DX12+DY12*DY12;D13P:=DX13*DX13+DY13*DY13;
  PA1:=D1*D1+D13P-D3*D3;PA2:=D1*D1+D12P-D2*D2;
  XP:=X1+(DY12*PA1-DY13*PA2)/AUX;
  YP:=Y1+(DX13*PA2-DX12*PA1)/AUX;
  SEEK(C,I);READ(C,LC);
  LC.X:=XP;LC.Y:=YP;SEEK(C,I);WRITE(C,LC);
  WRITELN('PUNCT ',PCTN);
  WRITELN('   X=',XP:12:4,'   Y=',YP:12:4);
 END ELSE IN_C_PR(I);
END; }

PROCEDURE INT_DIR_DIS;
VAR NRP1,NRP2,NRP3:LONGINT;
    ALFA1,ALFA2,ARG,AUX,S,X3,Y3,DIF:REAL;
    PCT1,PCT2,PCT3:STRING[6];
BEGIN
 NRP1:=0;NRP2:=0;NRP3:=0;XP:=0;YP:=0;J:=0;DIF:=0;I1:=0;
 X1:=0;Y1:=0;X2:=0;Y2:=0;X3:=0;Y3:=0;
 REPEAT
  J:=J+1; SEEK (D,J); READ (D,LD);
  IF (LD.TP1=NRP) OR (LD.TP2=NRP) THEN BEGIN
  IF NRP1=0 THEN BEGIN
    IF LD.TP1=NRP THEN NRP1:=LD.TP2;
    IF LD.TP2=NRP THEN NRP1:=LD.TP1;
    SEEK (C,NRP1); READ (C,LC);
    IF (LC.X<>0) THEN BEGIN
     PCT1:=LC.CODP;X1:=LC.X;Y1:=LC.Y;D1:=LD.DMR;I1:=I1+1;
    END ELSE BEGIN NRP1:=0;D1:=0; END;
  END;
  IF (NRP2=0) THEN BEGIN
    IF LD.TP1=NRP THEN NRP2:=LD.TP2;
    IF LD.TP2=NRP THEN NRP2:=LD.TP1;
    IF NRP2<>NRP1 THEN BEGIN
     SEEK (C,NRP2); READ (C,LC);
```

```
  IF (NRP3=0) THEN BEGIN
    IF LD.TP1=NRP THEN NRP3:=LD.TP2;
    IF LD.TP2=NRP THEN NRP3:=LD.TP1;
    IF (NRP3<>NRP1) AND (NRP3<>NRP2) THEN BEGIN
     SEEK (C,NRP3); READ (C,LC);
     IF (LC.X<>0) THEN BEGIN
      X3:=LC.X;Y3:=LC.Y;D3:=LD.DMR;PCT3:=LC.CODP;I1:=I1+1;
     END ELSE BEGIN NRP3:=0;D3:=0; END;
    END ELSE NRP3:=0;
  END;
  END;
 UNTIL (J=NTL) OR (I1=3);
 IF (I1=3) THEN BEGIN
  DX:=X2-X1;DY:=Y2-Y1;AUX:=DX*DX+DY*DY;DB:=SQRT(AUX);
  S:=0.5*(D1+D2+DB);
  AUX:=(S-D1)*(S-DB)/(S*S-S*D2);
  IF AUX<0 THEN BEGIN WRITELN ('COORDONATE GRESITE !');
  HALT; END;
  ARG:=SQRT(AUX);ALFA1:=2*(CARCTAN(ARG)); IF ALFA1>200
  THEN ALFA1:=400-ALFA1;
  AUX:=(S-D2)*(S-DB)/(S*S-S*D1);
  IF AUX<0 THEN BEGIN WRITELN ('COORDONATE GRESITE !');
  HALT; END;
  ARG:=SQRT(AUX);ALFA2:=2*(CARCTAN(ARG));
  IF ALFA2>200 THEN ALFA2:=400-ALFA2;
  ARG:=CTAN(ALFA1)+CTAN(ALFA2);
  XP:=(X1*CTAN(ALFA1)+X2*CTAN(ALFA2)+(Y2-Y1)*CTAN(ALFA1)*
  CTAN(ALFA2))/ARG;
  YP:=(Y1*CTAN(ALFA1)+Y2*CTAN(ALFA2)+(X1-X2)*CTAN(ALFA1)*
  CTAN(ALFA2))/ARG;
  DX:=XP-X3;DY:=YP-Y3;AUX:=DX*DX+DY*DY;D IF:=SQRT(AUX)-D3;
  IF ABS(DIF)>10 THEN BEGIN
   XP:=(X1*CTAN(ALFA1)+X2*CTAN(ALFA2)+(Y1-Y2)*CTAN(ALFA1)*
   CTAN(ALFA2))/ARG;
   YP:=(Y1*CTAN(ALFA1)+Y2*CTAN(ALFA2)+(X2-X1)*CTAN(ALFA1)*
   CTAN(ALFA2))/ARG;
   DX:=XP-X3;DY:=YP-Y3;AUX:=DX*DX+DY*DY;DIF:=SQRT(AUX)-D3;
  END;
  SEEK (C,I); READ (C,LC);LC.X:=XP;LC.Y:=YP;
  SEEK (C,I); WRITE (C,LC);
  WRITELN ('PUNCT ',PCTN);WRITELN ('   X=',XP:12:4,'   Y=',
  YP:12:4);
 END ELSE IN_C_PR(I);
```

```
 REP:=0;
 REPEAT
  REP:=REP+1;
  FOR I:=NPV+1 TO NPT DO BEGIN
   SEEK (C,I); READ (C,LC);
   IF (LC.X=0) THEN BEGIN
    NRP:=I;PCTN:=LC.CODP;INT_DIR_DIS;
   END;
  END;P_N_F_C;
 UNTIL (NPFC=0) OR (REP=2*NPT);
END;

PROCEDURE TRANSD;
BEGIN
 FOR I:=1 TO NTL DO BEGIN
  SEEK (D,I); READ (D,LD);LD.DMR:=LD.DM;LD.CRED:=0;
  SEEK (D,I); WRITE (D,LD);
 END;
END;

PROCEDURE INTRALT;
VAR ALT:REAL;
BEGIN
  TEXTCOLOR(RED); WRITE ('INTRODUCI ALTITUDINI ELIPSOIDICE
  MEDII ','ALE LATURILOR MASURATE ? D/N');
  TEXTCOLOR(BLUE);RASP;
  IF Q='D' THEN FOR I:=1 TO NTL DO BEGIN
    SEEK (D,I); READ (D,LD); WRITE ('LATURA ',LD.CODS,'-',
    LD.CODV);
    WRITE ('  ALTIT. ELIPS. MEDIE=');READLN(ALT);
    LD.TP3:=ROUND(ALT);
    SEEK (D,I); WRITE (D,LD);
  END ELSE BEGIN
    TEXTCOLOR(RED); WRITE ('INTRODUCI ALTITUDINEA ELIPSOIDICA
    MEDIE ','A INTREGII ZONE DE MASURATORI ? D/N');
    TEXTCOLOR(BLUE);RASP;
    IF Q='D' THEN BEGIN
      SEEK (D,0); READ (D,LD);
      WRITE ('ALTIT. ELIPS. MEDIE A ZONEI DE MASURATORI
      ESTE:');
      READLN(ALT);LD.TP3:=ROUND(ALT); SEEK (D,0);
      WRITE (D,LD);
    END;
```

```
BEGIN
  ALT:=0;ALT1:=0; SEEK (D,0); READ (D,LD);ALT:=LD.TP3;
  IF ALT=0 THEN BEGIN
    SEEK (D,1); READ (D,LD);ALT1:=LD.TP3;
  END;
  IF (ALT=0) AND (ALT1=0) THEN INTRALT;
  SEEK (D,0); READ (D,LD);ALT:=LD.TP3;
  SEEK (D,1); READ (D,LD);ALT1:=LD.TP3;
  IF (ALT=0) AND (ALT1=0) THEN
  WRITELN ('NU SUNT ELEMENTE SUFICIENTE PENTRU CALCUL
  CORECTII !');
  IF (ALT<>0) OR (ALT1<>0) THEN
  FOR I:=1 TO NTL DO BEGIN
    SEEK (D,I); READ (D,LD);
    SEEK (C,LD.TP1); READ (C,LC);X1:=LC.X;Y1:=LC.Y;
    SEEK (C,LD.TP2); READ (C,LC);X2:=LC.X;Y2:=LC.Y;C1:=0;
    IF ALT<>0 THEN C1:=1-ALT/6378956
    ELSE IF ALT1<>0 THEN C1:=1-LD.TP3/6378956;
    XM:=(X1+X2)/2000-500;YM:=(Y1+Y2)/2000-500;
    DX:=(X2-X1)/1000;DY:=(Y2-Y1)/1000;C2:=0;
    C2:=0.99975*(1+(XM*XM+YM*YM)/4/6379/6379-
    (DX*DX+DY*DY)/48/6379/6379);
    LD.CRED:=LD.DM*(1-C1*C2);LD.DMR:=LD.DM*C1*C2;
    SEEK (D,I); WRITE (D,LD);
  END;
END;

PROCEDURE D_O_CD_TL;
VAR L1,L2:LONGINT;
BEGIN
 FOR I:=1 TO NTL DO BEGIN
  SEEK (D,I); READ (D,LD);L1:=LD.TP1;L2:=LD.TP2;
  SEEK (C,L1); READ (C,LC);X1:=LC.X;Y1:=LC.Y;
  SEEK (C,L2); READ (C,LC);X2:=LC.X;Y2:=LC.Y;DX:=X2-X1;
  DY:=Y2-Y1;
  LD.D0:=SQRT(DX*DX+DY*DY);LD.T0:=TETA(DX,DY);LD.L0:=LD.D0-
  LD.DMR;
  LD.A0:=COS(LD.T0*PI/200);LD.B0:=SIN(LD.T0*PI/200);
  SEEK (D,I); WRITE (D,LD);
 END;
END;

PROCEDURE ME_ZERO(VAR E:FISR);
```

```
  POZ:=(I-1)*NTCOL+J;EE:=0; SEEK (E,POZ); WRITE (E,EE);
 END;
END;

PROCEDURE M_E_I;
VAR NTCOL,POZ,L1,L2:LONGINT;
    EE:REAL;
BEGIN
 NTCOL:=2*NPN+1;ME_ZERO(EI);ME_ZERO(ET);
 FOR I:=1 TO NTL DO BEGIN
  SEEK (D,I); READ (D,LD);L1:=LD.TP1;L2:=LD.TP2;
  POZ:=(I-1)*NTCOL+NTCOL;EE:=LD.L0;
  SEEK (EI,POZ); WRITE (EI,EE); SEEK (ET,POZ); WRITE (ET,EE);
  IF L1>NPV THEN BEGIN
   J1:=L1-NPV;POZ:=(I-1)*NTCOL+2*J1-1;EE:=-LD.A0;
   SEEK (EI,POZ); WRITE (EI,EE); SEEK (ET,POZ);
   WRITE (ET,EE);
   POZ:=POZ+1;EE:=-LD.B0;
   SEEK (EI,POZ); WRITE (EI,EE); SEEK (ET,POZ);
   WRITE (ET,EE);
   IF L2>NPV THEN BEGIN
    K1:=L2-NPV;POZ:=(I-1)*NTCOL+2*K1-1;EE:=LD.A0;
    SEEK (EI,POZ); WRITE (EI,EE); SEEK (ET,POZ);
    WRITE (ET,EE);
    POZ:=POZ+1;EE:=LD.B0;
    SEEK (EI,POZ); WRITE (EI,EE); SEEK (ET,POZ);
    WRITE (ET,EE);
   END;
  END;
  IF L1<=NPV THEN IF L2>NPV THEN BEGIN
   K1:=L2-NPV;POZ:=(I-1)*NTCOL+2*K1-1;EE:=LD.A0;
   SEEK (EI,POZ); WRITE (EI,EE); SEEK (ET,POZ);
   WRITE (ET,EE);
   POZ:=POZ+1;EE:=LD.B0;
   SEEK (EI,POZ); WRITE (EI,EE); SEEK (ET,POZ);
   WRITE (ET,EE);
  END;
 END;
END;

PROCEDURE M_E_T;
VAR NTCOL,POZ:LONGINT;
    EE:REAL;
```

```
   POZ:=(I-1)*NTCOL+J; SEEK (ET,POZ); READ (ET,EE);
   IF EE<>0 THEN
    IF LD.P0<>1 THEN BEGIN
     EE:=EE*SQRT(LD.P0); SEEK (ET,POZ); WRITE (ET,EE);
    END;
  END;
 END;
END;

PROCEDURE S_L;
VAR NTCOL,L,L1,L2,POZ:LONGINT;
    E1,E2:REAL;
    AA:DOUBLE;
BEGIN
 NTCOL:=2*NPN+1;
 FOR I:=1 TO NTCOL-1 DO FOR J:=I TO NTCOL DO BEGIN
  AA:=0;
  FOR K:=1 TO NTL DO BEGIN
   POZ:=(K-1)*NTCOL+I; SEEK (ET,POZ); READ (ET,E1);
   POZ:=(K-1)*NTCOL+J; SEEK (ET,POZ); READ (ET,E2);
   AA:=AA+E1*E2;
  END;
  IF J<NTCOL THEN BEGIN
   POZ:=(I-1)*2*NPN+J; SEEK (AN,POZ); WRITE (AN,AA);
   SEEK (A,POZ); WRITE (A,AA);
   POZ:=(J-1)*2*NPN+I; SEEK (AN,POZ); WRITE (AN,AA);
   SEEK (A,POZ); WRITE (A,AA);
  END;
  IF J=NTCOL THEN BEGIN
   AA:=-AA; SEEK (BN,I); WRITE (BN,AA); SEEK(B,I);
   WRITE (B,AA);
  END;
 END;
END;

PROCEDURE AFREZ;
BEGIN
 REPEAT
  TEXTCOLOR(BLUE);SCR_3;SCR_4;SCR_6;SCR_7;
  MS:='REPETI AFISAREA ? D/N';MESJR(50,WHEREY,MS);
 UNTIL Q='N';
END;
```

```
 REPEAT
  MSGX:='MATRICEA ECUATIILOR INITIALE';MSR;
  IF Q='D' THEN AFIS_M_E_IT(MSGX,EI);
  MSGX:='MATRICEA ECUATIILOR TRANSFORMATE';MSR;
  IF Q='D' THEN AFIS_M_E_IT(MSGX,ET);
  MSGX:='SISTEMUL DE ECUATII NORMALE';MSR;
  IF Q='D' THEN BEGIN
   MSGX:='MATRICEA COEFICIENTILOR';AFIS_MAT(MSGX,N,AN);
   MSGX:='VECTOR TERMENI LIBERI';AFIS_VEC(MSGX,N,BN);
  END;
  MSGX:='SISTEMUL DE ECUATII REDUS';MSR;
  IF Q='D' THEN BEGIN
   MSGX:='MATRICE REDUSA';AFIS_MAT(MSGX,N,A);
   MSGX:='VECTOR REDUS';AFIS_VEC(MSGX,N,B);
  END;
  MSGX:='MATRICEA COEFICIENTILOR DE PONDERE';MSR;
  IF Q='D' THEN AFIS_MAT(MSGX,N,A1);
  MS:='REPET AFISAREA ? D/N';MESJR(50,WHEREY,MS);
 UNTIL Q='N';
END;

PROCEDURE TIPREZ;
BEGIN
 REPEAT
  MS:='TIPARESTI COORDONATELE COMPENSATE SI PRECIZIILE ?D/N';
  MESJR(1,WHEREY,MS);
  IF Q='D' THEN BEGIN TESTTIP;TIP_6;TESTTIP;TIP_7; END;
  MS:='TIPARESTI CALCULELE DE COMPENSARE ? D/N';
  MESJR(1,WHEREY,MS);
  IF Q='D' THEN BEGIN TESTTIP;TIP_3; END;
  MS:='TIPARESTI CALCULELE DE VERIFICARE ? D/N';
  MESJR(1,WHEREY,MS);
  IF Q='D' THEN BEGIN TESTTIP;TIP_4; END;
  MS:='REPETI TIPARIREA ? D/N';MESJR(50,WHEREY,MS);
 UNTIL Q='N';
END;

PROCEDURE TIPREZ1;
VAR MSGX:STRING;
    NTCOL:LONGINT;
PROCEDURE MSR;
BEGIN MS:='TIPARESTI '+MSGX+' ? D/N';MESJR(1,WHEREY,MS); END;
BEGIN
```

```
  MSGX:='SISTEMUL DE ECUATII NORMALE';MSR;
  IF Q='D' THEN BEGIN
   MSGX:='MATRICE COEFICIENTI';TIP_MAT(MSGX,N,AN);
   MSGX:='VECTOR TERMENI LIBERI';TIP_VEC(MSGX,N,BN);
  END;
  MSGX:='SISTEMUL DE ECUATII REDUS';MSR;
  IF Q='D' THEN BEGIN
   MSGX:='MATRICE REDUSA';TIP_MAT(MSGX,N,A);
   MSGX:='VECTOR REDUS';TIP_VEC(MSGX,N,B);
  END;
  MSGX:='MATRICEA COEFICIENTILOR DE PONDERE';MSR;
  IF Q='D' THEN TIP_MAT(MSGX,N,A1);
  MS:='REPETI TIPARIREA ? D/N';MESJR(50,WHEREY,MS);
 UNTIL Q='N';
END;

PROCEDURE PREC;
VAR N,POZ:LONGINT;
    QXX,QXY,QYY: DOUBLE;
BEGIN
 N:=2*NPN;SEEK(C,0); READ (C,LC);EMP_UP:=SQRT(VV/(NTL2*NPN));
 LC.ET:=EMP_UP;SEEK(C,0); WRITE (C,LC);
 FOR I:=1 TO NPN DO BEGIN
  J1:=2*I-1;K1:=2*I;
  POZ:=(J1-1)*N+J1;SEEK(A1,POZ); READ (A1,QXX);
  SEEK(A1,(POZ+1)); READ (A1,QXY);
  POZ:=(K1-1)*N+K1;SEEK(A1,POZ); READ (A1,QYY);
  I1:=NPV+I;SEEK(C,I1); READ (C,LC);
  LC.EX:=EMP_UP*SQRT(QXX);LC.EY:=EMP_UP*SQRT(QYY);
  LC.ET:=SQRT(LC.EX*LC.EX+LC.EY*LC.EY);
  SEEK(C,I1); WRITE (C,LC);
 END;
END;

PROCEDURE CALCCOR;
VAR CONT,VER:INTEGER;
    S,EE,DIF:REAL;
    N,POZ:LONGINT;
    XX: DOUBLE;
    MSGX:STRING;
BEGIN
 FOR I:=1 TO NTL DO BEGIN
  SEEK(D,I); READ (D,LD);
```

```
END;
N:=2*NPN;
MSGX:='SE INTRODUC COORDONATE PROVIZORII ? D/N';
MESJR(1,WHEREY,MSGX);
IF Q='D' THEN BEGIN
  FOR I:=NPV+1 TO NPT DO BEGIN IN_C_PR(I);END;
END;
IF Q='N' THEN C_C_PR;
FOR I:=1 TO NPV DO BEGIN
 SEEK(C,I); READ (C,LC);LC.XC:=LC.X;LC.YC:=LC.Y;
 SEEK(C,I); WRITE (C,LC);
END;
MSGX:='CALCULEZ REDUCEREA DISTANTELOR LA PLANUL DE
PROIECTIE ? D/N';
MESJR(1,WHEREY,MSGX);
IF Q='D' THEN R_L_P ELSE TRANSD;
CONT:=0;
REPEAT
 VER:=0;CONT:=CONT+1;
 D_O_CD_TL;M_E_I;M_E_T;S_L;REDGAUSS(N,A,B);
 INV_MAT_TR_SUP(N,A,A1);PROD_MAT_VEC(N,A1,B,XM);
 FOR I:=NPV+1 TO NPT DO BEGIN
  SEEK(C,I); READ (C,LC);I1:=I-NPV;
  SEEK(XM,(2*I1-1)); READ (XM,XX);LC.XC:=LC.X+XX;LC.CX:=XX;
  SEEK(XM,(2*I1)); READ (XM,XX);LC.YC:=LC.Y+XX;LC.CY:=XX;
  SEEK(C,I); WRITE (C,LC);
 END;
 VV:=0;
 FOR I:=1 TO NTL DO BEGIN
  SEEK(D,I); READ (D,LD);S:=0;
  FOR J:=1 TO N DO BEGIN
   POZ:=(I-1)*(N+1)+J;
   SEEK(EI,POZ); READ (EI,EE);SEEK(XM,J); READ (XM,XX);
   S:=S+EE*XX;
  END;
  POZ:=I*(N+1);SEEK(EI,POZ); READ (EI,EE);
  LD.VI:=S+EE;LD.DD:=LD.DMR+LD.VI;
  SEEK(C,LD.TP1); READ (C,LC);X1:=LC.XC;Y1:=LC.YC;
  SEEK(C,LD.TP2); READ (C,LC);X2:=LC.XC;Y2:=LC.YC;
  DX:=X2-X1;DY:=Y2-Y1;LD.DC:=SQRT(DX*DX+DY*DY);
  SEEK(D,I); WRITE (D,LD);VV:=VV+LD.P0*LD.VI*LD.VI;
  DIF:=LD.DC-LD.DD;DIF:=ABS(DIF*1E5);
  IF DIF>1 THEN VER:=VER+1;
```

```
  END;
  IF NTL>2*NPN THEN PREC;
 UNTIL (VER=0) OR (CONT=1);
END;

PROCEDURE DFISMAT;
BEGIN
 FSPLIT(FIS1,DI,NUMFIS,EX);
 FIS:=CALE+NUMFIS+'.EI';ASSIGN(EI,FIS); REWRITE (EI);
 FIS:=CALE+NUMFIS+'.ET';ASSIGN(ET,FIS); REWRITE (ET);
 FIS:=CALE+NUMFIS+'.AN';ASSIGN(AN,FIS); REWRITE (AN);
 FIS:=CALE+NUMFIS+'.BN';ASSIGN(BN,FIS); REWRITE (BN);
 FIS:=CALE+NUMFIS+'.A';ASSIGN(A,FIS); REWRITE (A);
 FIS:=CALE+NUMFIS+'.B';ASSIGN(B,FIS); REWRITE (B);
 FIS:=CALE+NUMFIS+'.A1';ASSIGN(A1,FIS); REWRITE (A1);
 FIS:=CALE+NUMFIS+'.XM';ASSIGN(XM,FIS); REWRITE (XM);
END;

PROCEDURE SFISMAT;
BEGIN
 ERASE(A);ERASE(B);ERASE(A1);ERASE(XM);ERASE(EI);
 ERASE(ET);ERASE(AN);ERASE(BN);
END;

PROCEDURE TEXTREZ;
VAR MSGX:STRING;
BEGIN
  WRITE ('DORESTI SA CREEZI FISIERE TEXT CU REZULTATE ?D/N');
  RASP;
  IF Q='D' THEN BEGIN
   IF LENGTH(NUMFIS)<8 THEN FIS:='L'+NUMFIS;
   IF LENGTH(NUMFIS)<=8 THEN BEGIN
    FIS:=COPY(NUMFIS,2,7);FIS:='L'+FIS;
   END;
   FIS:=CALE+FIS+'.TXT';ASSIGN(F,FIS); REWRITE (F);
   TEXT_3;TEXT_4;TEXT_6;TEXT_7;
   MSGX:='MATRICEA ECUATIILOR INITIALE';
   Text_M_E_IT(MSGX,EI,F);
   MSGX:='MATRICEA ECUATIILOR TRANSFORMATE';
   Text_M_E_IT(MSGX,ET,F);
   MSGX:='SISTEMUL DE ECUATII NORMALE'; WRITELN (F,S0,MSGX);
   MSGX:='MATRICE COEFICIENTI';Text_MAT(MSGX,N,AN,F);
   MSGX:='VECTOR TERMENI LIBERI';Text_VEC(MSGX,N,BN,F);
```

```
   CLOSE(F); WRITELN ('AM REALIZAT FISIERUL TEXT ',FIS);
  END;
END;

BEGIN
 PGSTART;HIGHVIDEO;
 TEXTBACKGROUND(LIGHTGRAY);TEXTCOLOR(RED);CLRSCR; WRITELN;
 GOTOXY(23,5); WRITELN ('CALCUL-COMPENSARE RETEA
 TRILATERATIE');
 GOTOXY(25,WHEREY); WRITELN ('-metoda mas. indirecte
 ponderate-');
 WRITELN;TEXTCOLOR(BLUE); WRITE ('                  ');
 WRITELN ('Program realizat de Conf.dr.ing. Popia Adrian');
 WRITE ('                  ');
 WRITELN ('Catedra de Cadastru-Universitatea Tehnica Iasi');
 TEXTCOLOR(YELLOW); WRITELN;GOTOXY(27,WHEREY);
 WRITELN ('SISTEM DE COORDONATE X=NORD');
 GOTOXY(20,WHEREY); WRITELN ('UNITATE UNGHIURI  :
 GRAD CENTEZIMAL(GON)');
 GOTOXY(20,WHEREY); WRITELN ('     UNITATE DISTANTE  :
 METRU');
 TEXTCOLOR(BLUE); WRITELN;TEXTCOLOR(RED);
 GOTOXY(31,WHEREY);
 WRITELN ('DATELE SE INTRODUC');TEXTCOLOR(BLUE);
 WRITELN;GOTOXY(31,WHEREY); WRITELN ('1. DE LA TASTATURA');
 GOTOXY(31,WHEREY); WRITELN ('2. DE PE DISC');
 WRITE ('OPTIUNEA ESTE(1..2):');
 REPEAT SUN(1000,50);Q:= READKEY;
 UNTIL (Q='1') OR (Q='2');
 WRITELN (' ',Q);FI:=PARAMSTR(0);FSPLIT(FI,DI,NM,EX);
 IF NM<>'CTRIL99' THEN BEGIN ASSIGN(F,FI);ERASE(F);HALT; END;
 PAR:=' '; WRITE ('SCRIE PAROLA:');SUNG;TEXTCOLOR(GREEN);
 READLN(S0);
 IF S0<>PAR THEN BEGIN ASSIGN(F,FI);ERASE(F); END
 ELSE BEGIN
  TEXTCOLOR(BLUE);
  IF Q='1' THEN BEGIN
   C_T_1;C_T_2;IN_P;S_D;
   MS:='VERIFICI DATELE DE INTRARE ? D/N';MESJR(1,WHEREY,MS);
   IF Q='D' THEN  V_D;
   MS:='SE REINTRODUC PONDERILE ? D/N';MESJR(1,WHEREY,MS);
   IF Q='D' THEN IN_P;
   DFISMAT;CALCCOR;
```

```
   IF Q='D' THEN  V_D;
   MS:='SE REIA CALCULUL DE COMPENSARE ? D/N';
   MESJR(1,WHEREY,MS);
   IF Q='D' THEN BEGIN
    MS:='SE REINTRODUC PONDERILE ? D/N';MESJR(1,WHEREY,MS);
    IF Q='D' THEN IN_P;
    DFISMAT;CALCCOR;
    AFREZ;TIPREZ;AFREZ1;TIPREZ1;
   END;
  END;
  TEXTREZ;SFISMAT;
  WRITELN; WRITE ('APASA O TASTA PT. REVENIRE LA SISTEM !');
  SUN(1000,200);Q:= READKEY;
 END;
END.



{Turbo Pascal Runtime Library
 CRT Interface Unit
 Copyright (C) 1988,92 Borland International
 Modified(-> MyCrt Unit) in order to work on Win98,WinME}

UNIT MyCrt;

{$I-,S-}

INTERFACE

CONST
{CRT modes}
 BW40 = 0;            { 40x25 B/W on Color Adapter }
 CO40 = 1;            { 40x25 Color on Color Adapter }
 BW80 = 2;            { 80x25 B/W on Color Adapter }
 CO80 = 3;            { 80x25 Color on Color Adapter }
 Mono = 7;            { 80x25 on Monochrome Adapter }
 Font8x8= 256;        { Add-in for ROM font }

{Mode constants for 3.0 compatibility}
 C40           = CO40;
 C80           = CO80;

{Foreground and background color constants}
```

```
 Magenta       = 5;
 Brown         = 6;
 LightGray     = 7;

{Foreground color constants}
 DarkGray      = 8;
 LightBlue     = 9;
 LightGreen    = 10;
 LightCyan     = 11;
 LightRed      = 12;
 LightMagenta  = 13;
 Yellow        = 14;
 White         = 15;

{Add-in for blinking}
 Blink         = 128;

var
{Interface variables}
 CheckBreak:Boolean;    {Enable Ctrl-Break}
 CheckEOF:Boolean;      {Enable Ctrl-Z}
 DirectVideo:Boolean;   {Enable direct video addressing}
 CheckSnow:Boolean;     {Enable snow filtering}
 LastMode:Word;         {Current text mode}
 TextAttr:Byte;         {Current text attribute}
 WindMin:Word;          {Window upper left coordinates}
 WindMax:Word;          {Window lower right coordinates}

{Interface procedures}
procedure AssignCrt(var F:Text);
FUNCTION KeyPressed:Boolean;
FUNCTION READKEY:Char;
procedure TextMode(Mode:Integer);
procedure Window(X1,Y1,X2,Y2:Byte);
procedure GotoXY(X,Y:Byte);
FUNCTION WhereX:Byte;
FUNCTION WhereY:Byte;
procedure ClrScr;
procedure ClrEol;
procedure InsLine;
procedure DelLine;
procedure TextColor(Color:Byte);
procedure TextBackground(Color:Byte);
```

```
procedure NoSound;

IMPLEMENTATION

{$IFDEF DPMI}
{$L myCRT.OBP}
{$ELSE}
{$L myCRT.OBJ}
{$ENDIF}

procedure Initialize; External;
procedure AssignCrt; External;
FUNCTION KeyPressed; External;
FUNCTION READKEY; External;
procedure TextMode; External;
procedure Window; External;
procedure GotoXY; External;
FUNCTION WhereX; External;
FUNCTION WhereY; External;
procedure ClrScr; External;
procedure ClrEol; External;
procedure InsLine; External;
procedure DelLine; External;
procedure TextColor; External;
procedure TextBackground; External;
procedure LowVideo; External;
procedure HighVideo; External;
procedure NormVideo; External;
procedure Delay; External;
procedure Sound; External;
procedure NoSound; External;

procedure Break;
BEGIN
 Halt(255);
END;

BEGIN
 Initialize;
 AssignCrt(Input);
 Reset(Input);
 AssignCrt(Output);
 REWRITE (Output);
```

```
INTERFACE

USES GRAPH;
PROCEDURE ATTDRV;
PROCEDURE CGADRV;
PROCEDURE EGAVGADRV;
PROCEDURE HERCDRV;
PROCEDURE PC3270DRV;
PROCEDURE IBM8514DRV;

IMPLEMENTATION

PROCEDURE ATTDRV; External;
{$L ATT.OBJ }

PROCEDURE CGADRV; External;
{$L CGA.OBJ }

PROCEDURE EGAVGADRV; External;
{$L EGAVGA.OBJ }

PROCEDURE HERCDRV; External;
{$L HERC.OBJ }

PROCEDURE PC3270DRV; External;
{$L PC3270.OBJ }

PROCEDURE IBM8514DRV; External;
{$L IBM8514.OBJ }
END.

UNIT BGIFNT1;

INTERFACE
USES GRAPH;
PROCEDURE GOTHFNT;
PROCEDURE SANSFNT;
PROCEDURE LITTFNT;
PROCEDURE TRIPFNT;

IMPLEMENTATION
```

```
PROCEDURE LITTFNT; External;
{$L LITT.OBJ }

PROCEDURE TRIPFNT; External;
{$L TRIP.OBJ }

END.

UNIT BGIFNT2;

INTERFACE
USES GRAPH;
PROCEDURE EUROFNT;
PROCEDURE LCOMFNT;
PROCEDURE SIMPFNT;
PROCEDURE SCRIFNT;
PROCEDURE TSCRFNT;

IMPLEMENTATION

PROCEDURE EUROFNT; External;
{$L EURO.OBJ }

PROCEDURE LCOMFNT; External;
{$L LCOM.OBJ }

PROCEDURE SIMPFNT; External;
{$L SIMP.OBJ }

PROCEDURE SCRIFNT; External;
{$L SCRI.OBJ }

PROCEDURE TSCRFNT; External;
{$L TSCR.OBJ }
END.


UNIT FUNC;

INTERFACE

USES MYCRT;
```

```
{Calculeaza tangenta pentru un unghi exprimat in goni}
FUNCTION CARCSIN(VAR VAL:REAL):REAL;
{Calculeaza arcsinus pt. o valoare reala, cu rezultat in
goni}
FUNCTION CARCCOS(VAR VAL:REAL):REAL;
{Calculeaza arccosinus pt. o valoare reala, cu rezultat in
goni}
FUNCTION CARCTAN(VAR VAL:REAL):REAL;
{Calculeaza arctangent pt. o valoare reala, cu rezultat in
goni}
FUNCTION TETA(DX,DY:REAL):REAL;
{Calculeaza orientarea topografica in goni pt. raportul
dy/dx,X=NORD}
FUNCTION CN(CAR:CHAR;NRCAR:BYTE):STRING;
{Creaza un sir de NRCAR caractere de tipul dorit CAR}

IMPLEMENTATION

FUNCTION CSIN(VAR UNGHI:REAL):REAL;
BEGIN CSIN:=SIN(UNGHI*PI/200); END;

FUNCTION CCOS(VAR UNGHI:REAL):REAL;
BEGIN CCOS:=COS(UNGHI*PI/200); END;

FUNCTION CTAN(VAR UNGHI:REAL):REAL;
BEGIN
  IF (UNGHI<>100) AND (UNGHI<>300) THEN
  CTAN:=SIN(UNGHI*PI/200)/COS(UNGHI*PI/200);
END;

FUNCTION CARCSIN(VAR VAL:REAL):REAL;
VAR VAL2,ARGM:REAL;
BEGIN
  IF VAL=-1 THEN CARCSIN:=300;
  IF VAL=1  THEN CARCSIN:=100;
  IF (VAL<-1) OR (VAL>1) THEN BEGIN
   WRITELN ('ARGUMENT INCORECT IN FUNCTIA CARCSIN !');HALT;
  END;
  IF (VAL>-1) AND (VAL<1) THEN BEGIN
   VAL2:=1-VAL*VAL;ARGM:=VAL/SQRT(VAL2);
   CARCSIN:=(ARCTAN(ARGM))*200/PI;
  END;
END;
```

```
  IF VAL=1 THEN CARCCOS:=0.0;
  IF VAL=0 THEN CARCCOS:=100.0;
  IF (VAL<-1) OR (VAL>1) THEN BEGIN
   WRITELN ('ARGUMENT INCORECT IN FUNCTIA CARCCOS !');HALT;
  END;
  IF ((VAL>-1) AND (VAL<1)) AND (VAL<>0) THEN BEGIN
   VAL2:=VAL*VAL;ARGM:=(SQRT(1-VAL2))/VAL;
   VAL:=ARCTAN(ARGM)*200/PI; IF VAL<0 THEN VAL:=VAL+200;
   CARCCOS:=VAL;
  END;
END;

FUNCTION CARCTAN(VAR VAL:REAL):REAL;
BEGIN CARCTAN:=(ARCTAN(VAL))*200/PI; END;

FUNCTION CN(CAR:CHAR;NRCAR:BYTE):STRING;
VAR K:BYTE;
    SIR:STRING;
BEGIN
 SIR:=''; FOR K:=1 TO NRCAR DO SIR:=SIR+CAR;CN:=SIR;
END;

FUNCTION TETA(DX,DY:REAL):REAL;
VAR T:REAL;
BEGIN
 IF DX=0 THEN BEGIN
  IF DY=0 THEN BEGIN
   WRITELN ('ATENTIE,PUNCTE COINCIDENTE !');
   WRITELN ('NU SE POATE CALCULA ORIENTAREA !');
   REPEAT UNTIL KEYPRESSED;HALT
  END ELSE IF DY<0 THEN TETA:=300 ELSE TETA:=100
 END ELSE T:=(ARCTAN(DY/DX))*200/PI;
 IF DX<0 THEN BEGIN
  IF DY=0 THEN TETA:=200 ELSE TETA:=T+200;
 END;
 IF DX>0 THEN BEGIN
  IF DY=0 THEN TETA:=0 ELSE IF DY<0 THEN TETA:=T+400
  ELSE TETA:=T;
 END;
END;

BEGIN
END.
```

```
UNIT MES;

INTERFACE

USES MYCRT,PRINTER;
CONST MSG1:STRING='APASA O TASTA !';
      MSG2:STRING='COALA NOUA SI APASA O TASTA !';
      MSG3:STRING='ESTE GRESIT ??  D/N';
VAR Q:CHAR;
    COL:BYTE;
PROCEDURE SUN(FREC,DUR:INTEGER);
{Produce emisia unui sunet de frecventa FREC(Herti),
pe durata DUR(milisecunde)}
PROCEDURE SUNG;
{Produce emisia arpegiului}
PROCEDURE RASP;
{Emite un sunet si asteapta pina cind se apasa una din
tastele D sau N,apoi scrie tasta apasata}
PROCEDURE MESJ(PX,PY:BYTE;MSG:STRING);
{Realizeaza scrierea cu culoare rosu clipitor a mesajului MSG
pe ecran,pe linia PY,incepind de la coloana PX,emite doua
sunete,asteapta apasarea unei taste,apoi sterge mesajul si
lasa o linie blank}
PROCEDURE MESJ1(PX,PY:BYTE;MSG:STRING);
{Realizeaza scrierea cu culoare rosu neclipitor a mesajului
MSG pe ecran,pe linia PY,incepind de la coloana PX,emite doua
sunete,asteapta apasarea unei taste,apoi sterge mesajul si
lasa o linie blank}
PROCEDURE MESJR(PX,PY:BYTE;MSG:STRING);
{Realizeaza scrierea cu culoare rosu clipitor a mesajului MSG
pe ecran,pe linia PY,incepind de la coloana PX,emite doua
sunete,asteapta apasarea tastei D sau N,apoi sterge mesajul
si lasa o linie blank}
PROCEDURE TESTTIP;
{Testeaza daca imprimanta este in stare de functiune;
daca nu- afiseaza un mesaj si asteapta apasarea unei taste}

IMPLEMENTATION

PROCEDURE SUN(FREC,DUR:INTEGER);
BEGIN
  SOUND(FREC);DELAY(DUR);NOSOUND;
END;
```

```
BEGIN
  REPEAT
    SUN(900,50);Q:=UPCASE(READKEY);
  UNTIL (Q='D') OR (Q='N'); WRITELN (' ',Q);DELAY(200);
END;

PROCEDURE MESJ(PX,PY:BYTE;MSG:STRING);
VAR PPY:BYTE;
BEGIN
  PPY:=PY;GOTOXY(PX,PY);COL:=TEXTATTR;TEXTCOLOR(RED+BLINK);
  WRITE (MSG);SUN(523,50);SUN(1046,50);TEXTATTR:=COL;
  Q:= READKEY;GOTOXY(PX,PPY);CLREOL; WRITELN;GOTOXY(1,PPY);
  DELAY(200);
END;

PROCEDURE MESJ1(PX,PY:BYTE;MSG:STRING);
VAR PPY:BYTE;
BEGIN
  PPY:=PY;GOTOXY(PX,PY);COL:=TEXTATTR;TEXTCOLOR(RED);
  WRITE (MSG);
  SUN(523,50);SUN(1046,50);TEXTATTR:=COL;
  Q:= READKEY;GOTOXY(PX,PPY);CLREOL;
WRITELN;GOTOXY(1,PPY);DELAY(200);
END;

PROCEDURE MESJR(PX,PY:BYTE;MSG:STRING);
VAR PPY:BYTE;
BEGIN
  PPY:=PY;GOTOXY(PX,PY);COL:=TEXTATTR;TEXTCOLOR(RED); WRITE
(MSG);
  SUN(523,50);SUN(1046,50);TEXTATTR:=COL;
  RASP;GOTOXY(PX,PPY);CLREOL;
WRITELN;GOTOXY(1,PPY);DELAY(200);
END;

PROCEDURE TESTTIP;
BEGIN
    {$I-} WRITELN (LST);{$I+}
    IF IORESULT<>0 THEN BEGIN
      WRITELN ('ATENTIE, IMPRIMANTA NU ESTE IN REGULA !!');
      WRITELN ('REZOLVA PROBLEMA SI APASA O TASTA !!');
      SUN(1500,50);Q:= READKEY;
    END;
```

```
UNIT RSEND;

INTERFACE
{$N+}
USES MYCRT,PRINTER,MES;

TYPE FISD= FILE OF DOUBLE;
CONST NLP=64;NLE=22;
VAR  A,B,AN,BN,XM,A1:FISD;

PROCEDURE REDGAUSS(N:LONGINT;VAR M,V:FISD);
{Realizeaza reducerea unui sistem liniar de ecuatii normale
(N ec.cu N nec.) la forma superior triunghiulara prin metoda
Gauss cu pivotare.Matricea coeficientilor si vectorul
termenilor liberi sunt depuse in doua fisiere numerice in
dubla precizie care vor deveni elementele sistemului redus}
PROCEDURE INV_MAT_TR_SUP(N:LONGINT;VAR M,M1:FISD);
{Realizeaza inversarea unei matrici superior triunghiulare
 prin metoda Gauss-Jordan iar rezultatul este depus intr-o
 matrice separata;ambele matrici sunt fisiere numerice in
 dubla precizie}
PROCEDURE PROD_MAT_VEC(N:LONGINT;VAR M,V,X:FISD);
{Realizeaza produsul intre o matrice(N x N) si un vector(N)
iar rezultatul este depus intr-un vector(N) separat;se aplica
la rezolvarea sistemelor liniare cu matricea coeficientilor
inversata;matricea si cei doi vectori sunt fisiere numerice
in dubla precizie}
PROCEDURE RESIS_TR_SUP(N:LONGINT;VAR M,V,X:FISD);
{Realizeaza rezolvarea unui sistem liniar redus la forma
superior triunghiulara prin retrosubstitutie Gauss.Rezultatul
este depus in vectorul necunoscutelor;matricea si vectorii
sunt fisiere numerice in dubla precizie}
PROCEDURE MAT_ZERO(N:LONGINT;VAR M:FISD);
{Realizeaza initializarea cu zero a termenilor unei matici
patrate inregistrata ca fisier numeric in dubla precizie}
PROCEDURE TIP_SIS_LIN(VAR MSGX:STRING;N:LONGINT;VAR MA,MB:
FISD);
{Realizeaza tiparirea unui sistem liniar la imprimanta pe
coli A4; matricea coeficientilor si vectorul termeni liberi
sunt fisiere numerice dubla precizie}
PROCEDURE AFIS_SIS_LIN(VAR MSGX:STRING;N:LONGINT;VAR MA,MB:
FISD);
{Realizeaza afisarea unui sistem liniar pe monitor;matricea
```

```
{Realizeaza salvarea unui sistem liniar intr-un fisier text
al carui nume deriva din numele fisierelor de date;
 matricea coeficientilor si vectorul termeni liberi sunt
fisiere numerice dubla precizie}
PROCEDURE TIP_MAT(MSGX:STRING;N:LONGINT;VAR MA:FISD);
{Realizeaza tiparirea la imprimanta a unei matrici NxN
inregistrata ca fisier numeric in dubla precizie}
PROCEDURE AFIS_MAT(MSGX:STRING;N:LONGINT;VAR MA:FISD);
{Realizeaza afisarea pe monitor a unei matrici NxN
inregistrata ca fisier numeric in dubla precizie}
PROCEDURE Text_MAT(MSGX:STRING;N:LONGINT;VAR MA:FISD;VAR
F:Text);
{Realizeaza salvarea intr-un fisier text a unei matrici NxN
inregistrata ca fisier numeric in dubla precizie}
PROCEDURE TIP_VEC(MSGX:STRING;N:LONGINT;VAR VEC:FISD);
{Realizeaza tiparirea la imprimanta a unui vector inregistrat
 ca fisier numeric in dubla precizie}
PROCEDURE AFIS_VEC(MSGX:STRING;N:LONGINT;VAR VEC:FISD);
{Realizeaza afisarea pe monitor a unui vector inregistrat
ca fisier numeric in dubla precizie}
PROCEDURE Text_VEC(MSGX:STRING;N:LONGINT;VAR VEC:FISD;VAR
F:Text);
{Realizeaza salvarea intr-un fisier text a unui vector
inregistrat ca fisier numeric in dubla precizie}

IMPLEMENTATION

PROCEDURE REDGAUSS(N:LONGINT;VAR M,V:FISD);
VAR I,J,K,POZ:LONGINT;
    MJI,MII,MJK,MIK,T,VJ,VI: DOUBLE;
BEGIN
  FOR I:=1 TO N-1 DO
    FOR J:=I+1 TO N DO
      BEGIN
        POZ:=(J-1)*N+I;SEEK(M,POZ); READ (M,MJI);
        POZ:=(I-1)*N+I;SEEK(M,POZ); READ (M,MII);
        T:=MJI/MII;
        FOR K:=I TO N DO
          BEGIN
            POZ:=(J-1)*N+K;SEEK(M,POZ); READ (M,MJK);
            POZ:=(I-1)*N+K;SEEK(M,POZ); READ (M,MIK);
            MJK:=MJK-T*MIK;
            POZ:=(J-1)*N+K;SEEK(M,POZ); WRITE (M,MJK);
```

```
PROCEDURE INV_MAT_TR_SUP(N:LONGINT;VAR M,M1:FISD);
VAR I,J,K,POZ:LONGINT;
    SUM,AA,AA1: DOUBLE;
BEGIN
  FOR I:=N DOWNTO 1 DO
    BEGIN
      POZ:=(I-1)*N+I;SEEK(M,POZ); READ (M,AA);
      AA1:=1/AA;SEEK(M1,POZ); WRITE (M1,AA1);
      FOR J:=I-1 DOWNTO 1 DO
        BEGIN
          SUM:=0.0;
          FOR K:=J+1 TO I DO
            BEGIN
              POZ:=(J-1)*N+K;SEEK(M,POZ); READ (M,AA);
              POZ:=(K-1)*N+I;SEEK(M1,POZ); READ (M1,AA1);
              SUM:=SUM+AA*AA1;
            END;
          POZ:=(J-1)*N+J;SEEK(M,POZ); READ (M,AA);
          AA1:=-SUM/AA;
          POZ:=(J-1)*N+I;SEEK(M1,POZ); WRITE (M1,AA1);
        END;
      FOR J:=I+1 TO N DO
        BEGIN
          AA1:=0.0;
          POZ:=(J-1)*N+I;SEEK(M1,POZ); WRITE (M1,AA1);
        END;
    END;
END;

PROCEDURE PROD_MAT_VEC(N:LONGINT;VAR M,V,X:FISD);
VAR I,J,POZ:LONGINT;
    SS,AIJ,BJ: DOUBLE;
BEGIN
  FOR I:=1 TO N DO
    BEGIN
      SS:=0.00;
      FOR J:=1 TO N DO
        BEGIN
          POZ:=(I-1)*N+J;SEEK(M,POZ); READ (M,AIJ);
          SEEK(V,J); READ (V,BJ);SS:=SS+AIJ*BJ;
        END;
      SEEK(X,I); WRITE (X,SS);
```

```
    S,XI,XJ,AII,AIJ,BI: DOUBLE;
BEGIN
  XI:=0.00;
  FOR I:=1 TO N DO BEGIN SEEK(X,I); WRITE (X,XI); END;
  FOR I:=N DOWNTO 1 DO
    BEGIN
      S:=0.0;
      FOR J:=I+1 TO N DO
        BEGIN
          SEEK(X,J); READ (X,XJ);
          POZ:=(I-1)*N+J;SEEK(M,POZ); READ (M,AIJ);
          S:=S+AIJ*XJ;
        END;
      SEEK(V,I); READ (V,BI);POZ:=(I-1)*N+I;SEEK(M,POZ);
      READ (M,AII); XI:=(BI-S)/AII;SEEK(X,I); WRITE (X,XI);
    END;
END;

PROCEDURE MAT_ZERO(N:LONGINT;VAR M:FISD);
VAR I,J,POZ:LONGINT;
    AIJ: DOUBLE;
BEGIN
  FOR I:=1 TO N DO
    FOR J:=1 TO N DO
      BEGIN
        AIJ:=0.0;POZ:=(I-1)*N+J;
        SEEK(M,POZ); WRITE (M,AIJ);
      END;
END;

PROCEDURE TIP_SIS_LIN(VAR MSGX:STRING;N:LONGINT;VAR
MA,MB:FISD);
CONST S0:STRING[10]='          ';
VAR   I,J,I1,J1,L1,L2,POZ:LONGINT;
      M1,M2,K1,K2,K3:LONGINT;
      AIJ: DOUBLE;
BEGIN
  TESTTIP; WRITELN (LST,S0,MSGX);
  WRITELN (LST,' ');
  J1:=TRUNC((N+1)/4); IF (4*J1)<(N+1) THEN J1:=J1+1;
  IF J1=0 THEN J1:=1;
  I1:=TRUNC(N/NLP); IF (NLP*I1)<N THEN I1:=I1+1;
  IF I1=0 THEN I1:=1;
```

```
IF (L1>1) OR (L2>1) THEN MESJ(30,WHEREY,MSG2);
K1:=L2*4-3;K2:=L2*4;
WRITE (LST,S0,'      ');
FOR J:=K1 TO K2 DO
    IF J<=(N+1) THEN
      BEGIN
        IF J<(N+1) THEN
          BEGIN
            IF J<K2 THEN
            WRITE (LST,'       C ',J:6);
            IF J=K2 THEN
            WRITELN (LST,'       C',J:6);
          END;
        IF J=(N+1) THEN
        WRITELN (LST,'       T.LIB.');
      END; WRITELN (LST,' ');
FOR I:=M1 TO M2 DO
  BEGIN
    IF I<=N THEN
      BEGIN
        WRITE (LST,S0,'L ',I:4);
        FOR J:=K1 TO K2 DO
          BEGIN
            IF J<=(N+1) THEN
              BEGIN
                IF J<(N+1) THEN
                  BEGIN
                    POZ:=(I-1)*N+J;SEEK(MA,POZ);
                    READ (MA,AIJ);
                    IF J<K2 THEN IF
                    ABS(AIJ)>=0.000001 THEN
                    WRITE (LST,'  ',AIJ:13:6)
                    ELSE WRITE
                    (LST,'  ','      -      ');
                    IF J=K2 THEN IF
                    ABS(AIJ)>=0.000001 THEN
                    WRITELN (LST,'  ',AIJ:13:6)
                    ELSE WRITELN
                    (LST,'  ','      -      ');
                  END;
                IF J=(N+1) THEN
                  BEGIN
                    SEEK(MB,I); READ (MB,AIJ);
```

```
                        END;
                    END;
                END;
            END; WRITELN (LST,' ');MESJ(30,WHEREY,MSG2);
        END;
    END;
END;

PROCEDURE AFIS_SIS_LIN(VAR MSGX:STRING;N:LONGINT;VAR
MA,MB:FISD);
CONST S0:STRING='           ';
VAR   I,J,I1,J1,L1,L2,POZ:LONGINT;
      M1,M2,K1,K2,K3:LONGINT;
      AIJ: DOUBLE;COL:BYTE;
BEGIN
  COL:=TEXTATTR;TEXTCOLOR(LIGHTRED);
  WRITELN (S0,MSGX);TEXTATTR:=COL;
  J1:=TRUNC((N+1)/4); IF (4*J1)<(N+1) THEN J1:=J1+1;
  IF J1=0 THEN J1:=1;
  I1:=TRUNC(N/NLE); IF (NLE*I1)<N THEN I1:=I1+1;
  IF I1=0 THEN I1:=1;
  FOR L1:=1 TO I1 DO
    BEGIN
      M1:=L1*NLE-NLE+1;M2:=L1*NLE;
      FOR L2:=1 TO J1 DO
        BEGIN
          IF (L1>1) OR (L2>1) THEN
            BEGIN MESJ(60,WHEREY,MSG1); END;
          K1:=L2*4-3;K2:=L2*4;
          WRITE (S0,'      ');
          FOR J:=K1 TO K2 DO
              IF J<=(N+1) THEN
                BEGIN
                  IF J<(N+1) THEN
                    BEGIN
                      IF J<K2 THEN WRITE ('       C ',J:6);
                      IF J=K2 THEN WRITELN ('       C',J:6);
                    END;
                  IF J=(N+1) THEN WRITELN ('       T.LIB.');
                END; WRITELN (' ');
          FOR I:=M1 TO M2 DO
            BEGIN
              IF I<=N THEN
```

```
                              BEGIN
                                IF J<(N+1) THEN
                                  BEGIN
                                    POZ:=(I-1)*N+J;
                                    SEEK(MA,POZ); READ (MA,AIJ);
                                    IF J<K2 THEN IF
                                    ABS(AIJ)>=0.000001 THEN
                                    WRITE ('  ',AIJ:13:6)
                                    ELSE WRITE
                                    ('  ','      -      ');
                                    IF J=K2 THEN IF
                                    ABS(AIJ)>=0.000001 THEN
                                    WRITELN ('  ',AIJ:13:6)
                                    ELSE WRITELN
                                    ('  ','      -      ');
                                  END;
                                IF J=(N+1) THEN
                                  BEGIN
                                    SEEK(MB,I); READ (MB,AIJ);
                                    IF ABS(AIJ)>=0.000001 THEN
                                    WRITELN ('  ',AIJ:13:6)
                                    ELSE WRITELN
                                    ('  ','      -      ');
                                  END;
                              END;
                          END;
                      END;
                  END;MESJ(60,WHEREY,MSG1);
              END;
          END;
END;

PROCEDURE Text_SIS_LIN(VAR MSGX:STRING;N:LONGINT;VAR
MA,MB:FISD;VAR F:Text);
CONST S0:STRING[10]='          ';
VAR   I,J,I1,J1,L1,L2,POZ:LONGINT;
      M1,M2,K1,K2,K3:LONGINT;
      AIJ: DOUBLE;
BEGIN
  WRITELN (F,S0,MSGX);
  WRITELN (F,' ');
  J1:=TRUNC((N+1)/4); IF (4*J1)<(N+1) THEN J1:=J1+1;
  IF J1=0 THEN J1:=1;
```

```
FOR L2:=1 TO J1 DO
  BEGIN
    IF (L1>1) OR (L2>1) THEN WRITELN (F,S0);
    K1:=L2*4-3;K2:=L2*4;
    WRITE (F,S0,'      ');
    FOR J:=K1 TO K2 DO
        IF J<=(N+1) THEN
          BEGIN
            IF J<(N+1) THEN
              BEGIN
                IF J<K2 THEN WRITE(F,'       C ',J:6);
                IF J=K2 THEN WRITELN
                (F,'       C',J:6);
              END;
            IF J=(N+1) THEN WRITELN
            (F,'       T.LIB.');
          END; WRITELN (F,' ');
    FOR I:=M1 TO M2 DO
      BEGIN
        IF I<=N THEN
          BEGIN
            WRITE (F,S0,'L ',I:4);
            FOR J:=K1 TO K2 DO
              BEGIN
                IF J<=(N+1) THEN
                  BEGIN
                    IF J<(N+1) THEN
                      BEGIN
                        POZ:=(I-1)*N+J;SEEK(MA,POZ);
                        READ (MA,AIJ);
                        IF J<K2 THEN IF
                        ABS(AIJ)>=0.000001 THEN
                        WRITE (F,'  ',AIJ:13:6)
                        ELSE WRITE
                        F,'  ','      -      ');
                        IF J=K2 THEN IF
                        ABS(AIJ)>=0.000001 THEN
                        WRITELN (F,'  ',AIJ:13:6)
                        ELSE WRITELN
                        (F,'  ','      -      ');
                      END;
                    IF J=(N+1) THEN
                      BEGIN
```

```
                              END;
                          END;
                      END;
                  END;
              END; WRITELN (F,' ');
          END;
      END;
END;

PROCEDURE TIP_MAT(MSGX:STRING;N:LONGINT;VAR MA:FISD);
CONST S0:STRING='          ';
VAR   I,J,I1,J1,L1,L2,POZ:LONGINT;
      M1,M2,K1,K2,K3:LONGINT;
      AIJ: DOUBLE;
BEGIN
  TESTTIP; WRITELN (LST,S0,MSGX); WRITELN (LST,' ');
  J1:=TRUNC(N/4); IF (4*J1)<N THEN J1:=J1+1;
  IF J1=0 THEN J1:=1;
  I1:=TRUNC(N/NLP); IF (NLP*I1)<N THEN I1:=I1+1;
  IF I1=0 THEN I1:=1;
  FOR L1:=1 TO I1 DO
    BEGIN
      M1:=L1*NLP-NLP+1;M2:=L1*NLP;
      FOR L2:=1 TO J1 DO
        BEGIN
          IF (L1>1) OR (L2>1) THEN MESJ(30,WHEREY,MSG2);
          K1:=L2*4-3;K2:=L2*4;
          WRITE (LST,S0,'      ');
          FOR J:=K1 TO K2 DO
              IF J<=N THEN WRITE (LST,'       C ',J:6);
          WRITELN (LST,' '); WRITELN (LST,' ');
          FOR I:=M1 TO M2 DO
            BEGIN
              IF I<=N THEN
                BEGIN
                  WRITE (LST,S0,'L ',I:4);
                  FOR J:=K1 TO K2 DO
                      IF J<=N THEN
                        BEGIN
                          POZ:=(I-1)*N+J;SEEK(MA,POZ);
                          READ (MA,AIJ);
                          IF ABS(AIJ)>=0.000001 THEN
                          WRITE (LST,'  ',AIJ:13:6)
```

```
            END;WRITELN(LST,' ');
        END;
    END;
END;

PROCEDURE AFIS_MAT(MSGX:STRING;N:LONGINT;VAR MA:FISD);
CONST S0:STRING='          ';
VAR   I,J,I1,J1,L1,L2,POZ:LONGINT;
      M1,M2,K1,K2,K3:LONGINT;
      AIJ: DOUBLE;COL:BYTE;
BEGIN
  COL:=TEXTATTR;TEXTCOLOR(LIGHTRED);
  WRITELN(S0,MSGX);TEXTATTR:=COL;
  J1:=TRUNC(N/4); IF (4*J1)<N THEN J1:=J1+1;
  IF J1=0 THEN J1:=1;
  I1:=TRUNC(N/NLE); IF (NLE*I1)<N THEN I1:=I1+1;
  IF I1=0 THEN I1:=1;
  FOR L1:=1 TO I1 DO
    BEGIN
      M1:=L1*NLE-NLE+1;M2:=L1*NLE;
      FOR L2:=1 TO J1 DO
        BEGIN
          IF (L1>1) OR (L2>1) THEN
            BEGIN MESJ(60,WHEREY,MSG1);CLRSCR; END;
          K1:=L2*4-3;K2:=L2*4;
          WRITE (S0,'      ');
          FOR J:=K1 TO K2 DO
              IF J<=N THEN WRITE ('       C ',J:6);
          WRITELN (' ');
          FOR I:=M1 TO M2 DO
            BEGIN
              IF I<=N THEN
                BEGIN
                  WRITE (S0,'L ',I:4);
                  FOR J:=K1 TO K2 DO
                      IF J<=N THEN
                        BEGIN
                          POZ:=(I-1)*N+J;SEEK(MA,POZ);
                          READ (MA,AIJ);
                          IF ABS(AIJ)>=0.000001 THEN WRITE
                          ('  ',AIJ:13:6)
                          ELSE WRITE ('  ','      -      ');
                        END;
```

```
END;

PROCEDURE Text_MAT(MSGX:STRING;N:LONGINT;VAR MA:FISD;VAR
F:Text);
CONST S0:STRING='          ';
VAR   I,J,I1,J1,L1,L2,POZ:LONGINT;
      M1,M2,K1,K2,K3:LONGINT;
      AIJ: DOUBLE;
BEGIN
  WRITELN (F,S0,MSGX); WRITELN (F,' ');
  J1:=TRUNC(N/4); IF (4*J1)<N THEN J1:=J1+1;
  IF J1=0 THEN J1:=1;
  I1:=TRUNC(N/NLP); IF (NLP*I1)<N THEN I1:=I1+1;
  IF I1=0 THEN I1:=1;
  FOR L1:=1 TO I1 DO
    BEGIN
      M1:=L1*NLP-NLP+1;M2:=L1*NLP;
      FOR L2:=1 TO J1 DO
        BEGIN
          IF (L1>1) OR (L2>1) THEN WRITELN (F,S0);
          K1:=L2*4-3;K2:=L2*4;
          WRITE (F,S0,'      ');
          FOR J:=K1 TO K2 DO
              IF J<=N THEN WRITE (F,'       C ',J:6);
          WRITELN (F,' '); WRITELN (F,' ');
          FOR I:=M1 TO M2 DO
            BEGIN
              IF I<=N THEN
                BEGIN
                  WRITE (F,S0,'L ',I:4);
                  FOR J:=K1 TO K2 DO
                      IF J<=N THEN
                        BEGIN
                          POZ:=(I-1)*N+J;SEEK(MA,POZ);
                          READ (MA,AIJ);
                          IF ABS(AIJ)>=0.000001 THEN
                          WRITE (F,'  ',AIJ:13:6)
                          ELSE WRITE (F,'  ','      -     ');
                        END;
                      WRITELN (F,' ');
                END;
            END; WRITELN (F,' ');
        END;
```

```
VAR I,I1:LONGINT;
    BI: DOUBLE;
BEGIN
  TESTTIP; WRITELN (LST,S0,MSGX);I1:=0;
  FOR I:=1 TO N DO
    BEGIN
      SEEK(VEC,I); READ (VEC,BI);
      WRITELN (LST,S0,'T',I:3,'=',BI:13:6);
      I1:=I1+1; IF I1>=NLP THEN
      BEGIN MESJ(30,WHEREY,MSG2);I1:=0; END;
    END;
END;

PROCEDURE AFIS_VEC(MSGX:STRING;N:LONGINT;VAR VEC:FISD);
CONST S0:STRING='          ';
VAR I,I1:LONGINT;
    BI: DOUBLE;COL:BYTE;
BEGIN
  COL:=TEXTATTR;TEXTCOLOR(LIGHTRED); WRITELN (S0,MSGX);
  TEXTATTR:=COL;I1:=0;
  FOR I:=1 TO N DO
    BEGIN
      SEEK(VEC,I); READ (VEC,BI);
      WRITELN (S0,' T',I:3,'=',BI:13:6);
      I1:=I1+1; IF I1>=NLE THEN
      BEGIN MESJ(60,WHEREY,MSG1);I1:=0; END;
    END;
END;

PROCEDURE Text_VEC(MSGX:STRING;N:LONGINT;VAR VEC:FISD;VAR
F:Text);
CONST S0:STRING='          ';
VAR I,I1:LONGINT;
    BI: DOUBLE;
BEGIN
  WRITELN (F,S0,MSGX);I1:=0;
  FOR I:=1 TO N DO
    BEGIN
      SEEK(VEC,I); READ (VEC,BI);
      WRITELN (F,S0,'T',I:3,'=',BI:13:6);
      I1:=I1+1; IF I1>=NLP THEN
      BEGIN WRITELN (F,S0);I1:=0; END;
    END;
```

```
UNIT TTRIL;

INTERFACE
USES DOS,MYCRT,PRINTER,MES,FUNC;

TYPE LNC=RECORD
           PT,PV,PN,PS:INTEGER;
           CODP:STRING[6];
           X,Y,CX,CY,XC,YC:REAL;
           EX,EY,EA,EB,ET:REAL;
           SC1,SC2:LONGINT;
          END;
     LND= RECORD
            CODS,CODV:STRING[6];
            DM,CRED,DMR,D0,T0,P0:REAL;
            A0,B0,L0,VI,DC,DD:REAL;
            TP1,TP2,TP3:LONGINT;
           END;
      FISR= FILE OF REAL;
      FISD= FILE OF DOUBLE;
{ PV=total puncte vechi         }
{ PN=total puncte noi           }
{ CODP=nume punct din retea     }
{ X,Y=coordonate provizorii     }
{ CX,CY=corectii coordonate     }
{ XC,YC=coordonate compensate   }
{ EX,EY=erorile coordonatelor   }
{ ET=eroarea totala a punctului }
{ TP1=nr.pct.st. in fisierul de coordonate }
{ TP2=nr.pct.viz. in fisierul de coordonate }
{ CODS=nume punct de statie     }
{ CODV=nume punct vizat         }
{ DM=distante(laturi) masurate  }
{ D0,T0=dist. si or. provizorii }
{ L0=termeni liberi             }
{ DC=laturi(dist.) compensate   }
{ A0,B0=coef. ec. initiale      }
{ VI=corectii dist. masurate    }
{ DD=laturi(dist.) calc.din coord.}

CONST NLP=64;NLE=21;
VAR S0,S1,S2,S3,S4,S5,S6,S7,S8:STRING;
    LC:LNC;LD:LND;
```

```
    NM,NUMFIS:NAMESTR;DI,CALE:DIRSTR;EX,EXT:EXTSTR;
PROCEDURE CAP_T1;
{Cap tab introducere coord. puncte de sprijin trilat.}
PROCEDURE SCR_CAP_T1;
{Scriere pe ecran a lui CAP_T1}
PROCEDURE CAP_T2;
{Cap tab introducere laturi masurate in reteaua de trilat.}
PROCEDURE SCR_CAP_T2;
{Afisare CAP_T2}
PROCEDURE SCR_3;
{Afisare a calculelor de compensare a trilat.}
PROCEDURE TIP_3;
{Tiparire la impr. a calculelor de compensare a trilat.}
PROCEDURE SCR_4;
{Afisare a verificarilor compensarii trilat.}
PROCEDURE TIP_4;
{Tiparire a verificarilor compensarii trilat.}
PROCEDURE SCR_6;
{Afisare corectii si coord. compensate  trilat.}
PROCEDURE TIP_6;
{Tiparire corectii si coord. compensate  trilat.}
PROCEDURE SCR_7;
{Afisare erori coordonate}
PROCEDURE TIP_7;
{Tiparire erori coordonate}
PROCEDURE AFIS_M_E_IT(MSG:STRING;VAR E:FISR);
{Afisare matrice ecuatii initiale/transformate de erori}
{MSG=mesaj E=matrice cu elem.reale pe disc}
PROCEDURE TIP_M_E_IT(MSG:STRING;VAR E:FISR);
{Tiparire matrice ecuatii initiale/transformate de erori}
{MSG=mesaj E=matrice cu elem.reale pe disc}
PROCEDURE Text_M_E_IT(MSG:STRING;VAR E:FISR;VAR F:TEXT);
{Salvare intr-un fisier text a matricei ecuatiilor
initiale/transformate de erori}
{MSG=mesaj E=matrice cu elem.reale pe disc}
PROCEDURE TEXT_3; PROCEDURE TEXT_4; PROCEDURE TEXT_6;
PROCEDURE TEXT_7;
{Salveaza rezultatele compensarii intr-un fisier text}

IMPLEMENTATION
PROCEDURE CAP_T1;
BEGIN
{    25      33          45          57
```

```
S6:='|-------------------------------|';
S7:='|       |           |           |';
S8:='|-------------------------------|';}
  S0:='                          ';
  S1:=#201+CN(#205,7)+#209+CN(#205,23)+#187;
  S2:=#186+' NUME  '+#179+'   COORDONATE  PLANE   '+#186;
  S3:=#186+'       '+#195+CN(#196,11)+#194+CN(#196,11)+#182;
  S4:=#186+' PUNCT '+#179+'   Nord    '+#179+'    Est
      '+#186;
  S5:=#186+'       '+#179+'    (m)    '+#179+'    (m)
      '+#186;
  S6:=#204+CN(#205,7)+#216+CN(#205,11)+#216+CN(#205,11)+#185;
  S7:=#186+'       '+#179+'           '+#179+
  '           '+#186;
  S8:=#200+CN(#205,7)+#207+CN(#205,11)+#207+CN(#205,11)+#188;
END;

PROCEDURE SCR_CAP_T1;
BEGIN
 WRITELN (S0,S1); WRITELN (S0,S2); WRITELN (S0,S3);
 WRITELN (S0,S4);WRITELN (S0,S5); WRITELN (S0,S6);
END;

PROCEDURE CAP_T2;
BEGIN
{    26      34      42         53
S1:='|-------------------------|';
S2:='|LATURA MASURATA| DISTANTA |';
S3:='|---------------|ORIZONTALA|';
S4:='| DE LA |  LA   | MASURATA |';
S5:='|PUNCTUL|PUNCTUL|    (m)   |';
S6:='|-------------------------|';
S7:='|       |       |          |';
S8:='|-------------------------|';}
  S0:='                          ';
  S1:=#201+CN(#205,7)+#205+CN(#205,7)+#209+CN(#205,10)+#187;
  S2:=#186+'LATURA MASURATA'+#179+' DISTANTA '+#186;
  S3:=#199+CN(#196,7)+#194+CN(#196,7)+#180+'ORIZONTALA'+#186;
  S4:=#186+' DE LA '+#179+'   LA  '+#179+' MASURATE '+#186;
  S5:=#186+'PUNCTUL'+#179+'PUNCTUL'+#179+'   (m)    '+#186;
  S6:=#204+CN(#205,7)+#216+CN(#205,7)+#216+CN(#205,10)+#185;
  S7:=#186+'       '+#179+'       '+#179+'          '+#186;
  S8:=#200+CN(#205,7)+#207+CN(#205,7)+#207+CN(#205,10)+#188;
```

```
 WRITELN (S0,S4);WRITELN (S0,S5); WRITELN (S0,S6);
END;

PROCEDURE CAP_T3;
BEGIN
{    9      16     23         34        44        54
64       73
S1:='|-------------|----------|---------|-------------------
|--------|';
S2:='|LATURA RETEA | DISTANTA |ORIENTARE|   COEF.  ECUATII  |
TERMEN |';
S3:='|------|------|          |         |---------|---------
|        |';
S4:='| DE LA|  LA  |  PROVIZ. | PROVIZ. |    a    |    b
|  LIBER |';
S5:='|  PCT.|  PCT.|    (m)   |  (gon)  |         |
|   (m)  |';
S6:='|------|------|----------|---------|---------|---------
|--------|';
S7:='|      |      |          |         |         |
|        |';
S8:='|------|------|----------|---------|---------|---------
|--------|';}
  S0:='        ';
  S1:=#201+CN(#205,13)+#209+CN(#205,10)+#209+CN(#205,9)+#209+
      CN(#205,19)+#209+CN(#205,8)+#187;
  S2:=#186+'LATURA RETEA '+#179+' DISTANTA '+#179+'ORIENTARE'
      +#179+'   COEF.  ECUATII  '+#179+' TERMEN '+#186;
  S3:=#199+CN(#196,6)+#194+CN(#196,6)+#180+'          '+#179+
      '         '+#195+CN(#196,9)+#194+CN(#196,9)+#180+
      '        '+#186;
  S4:=#186+' DE LA'+#179+'  LA  '+#179+'  PROVIZ. '+#179+
      ' PROVIZ. '+#179+'    a    '+#179+'    b    '+#179+
      '  LIBER '+#186;
  S5:=#186+' PCT. '+#179+'  PCT.'+#179+'   (m)    '+#179+
      '  (gon)  '+
      #179+'         '+#179+'         '+#179+'   (m)  '+#186;
  S6:=#204+CN(#205,6)+#216+CN(#205,6)+#216+CN(#205,10)+
      #216+CN(#205,9)+#216+CN(#205,9)+#216+CN(#205,9)+#216+
      CN(#205,8)+#185;
  S7:=#186+'      '+#179+'      '+#179+'          '+#179+
      '         '+#179+'         '+#179+'         '+#179+
      '        '+#186;
```

```
PROCEDURE SCR_3;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE SCRCAP3;
BEGIN
 CAP_T3; WRITELN (S0,'CALCULE COMPENSARE LA LUCRAREA ',
 NUMFIS);
 WRITELN (S0,S1); WRITELN (S0,S2); WRITELN (S0,S3);
 WRITELN (S0,S4);WRITELN (S0,S5); WRITELN (S0,S6);NL:=7;
END;
BEGIN
  CLRSCR;SCRCAP3;
  FOR I:=1 TO NTL DO
    BEGIN
      SEEK(D,I); READ (D,LD);
      WRITE (S0,#186,LD.CODS:6,#179,LD.CODV:6,
             #179,LD.D0:10:4,#179);
      WRITELN (LD.T0:9:4,#179,LD.A0:9:6,#179,LD.B0:9:6,
               #179,LD.L0:8:4,#186);
      NL:=NL+1;
      IF NL>=NLE THEN IF I<NTL THEN
        BEGIN
          WRITELN (S0,S8);MESJ(60,WHEREY,MSG1);
          CLRSCR;SCRCAP3;
        END;
    END;
  WRITELN (S0,S8);MESJ(60,WHEREY,MSG1);
END;

PROCEDURE TIP_3;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE TIPCAP3;
BEGIN
 CAP_T3; WRITELN (LST,S0,'CALCULE COMPENSARE LA LUCRAREA ',
 NUMFIS);
 WRITELN (LST,S0,S1); WRITELN (LST,S0,S2);
 WRITELN (LST,S0,S3);
 WRITELN (LST,S0,S4); WRITELN (LST,S0,S5);
 WRITELN (LST,S0,S6);NL:=7;
END;
BEGIN
  CLRSCR;TIPCAP3;
```

```
      WRITELN (LST,LD.T0:9:4,#179,LD.A0:9:6,#179,LD.B0:9:6,
               #179,LD.L0:8:4,#186);
      NL:=NL+1;
      IF NL>=NLP THEN IF I<NTL THEN
        BEGIN
          WRITELN (LST,S0,S8);MESJ(60,WHEREY,MSG2);
          CLRSCR;TIPCAP3;
        END;
    END;
  WRITELN (LST,S0,S8);MESJ(60,WHEREY,MSG1);
END;

PROCEDURE CAP_T4;
BEGIN
{    13     20     27         38      44       53         64
75
S1:='|-------------|----------|-------|--------|----------|--
--------|';
S2:='|LATURA RETEA | DISTANTE |PONDERE|CORECTII| DISTANTE |
DISTANTE |';
S3:='|------|------|          | MASU- |DISTANTE|          |
DIN    |';
S4:='| DE LA| LA   | MASURATE | RARE  |  MAS.  | COMPENS.
|COORD.COMP|';
S5:='| PCT. | PCT. |    (m)   |       |   (m)  |    (m)   |
(m)   |';
S6:='|------|------|----------|-------|--------|----------|--
--------|';
S7:='|      |      |          |       |        |          |
|';
S8:='|------|------|----------|-------|--------|----------|--
--------|';}
  S0:='          ';
  S1:=#201+CN(#205,13)+#209+CN(#205,10)+#209+CN(#205,7)+#209+
      CN(#205,8)+#209+CN(#205,10)+#209+CN(#205,10)+#187;
  S2:=#186+'LATURA RETEA '+#179+' DISTANTE '+#179+'PONDERE'+
      #179+'CORECTII'+#179+' DISTANTE '+#179+' DISTANTE '+
      #186;
  S3:=#199+CN(#196,6)+#194+CN(#196,6)+#180+'          '+#179+
      ' MASU- '+#179+'DISTANTE'+#179+'          '+#179+
      '   DIN    '+#186;
  S4:=#186+' DE LA'+#179+' LA   '+#179+' MASURATE '+#179+
      ' RARE  '+#179+'  MAS.  '+#179+' COMPENS. '+#179+
```

```
      #216+CN(#205,7)+#216+CN(#205,8)+#216+CN(#205,10)+
      #216+CN(#205,10)+#185;
  S7:=#186+'      '+#179+'      '+#179+'          '+#179+
      '       '+#179+'        '+#179+'          '+#179+
       '          '+#186;
   S8:=#200+CN(#205,6)+#207+CN(#205,6)+#207+CN(#205,10)+
       #207+CN(#205,7)+#207+CN(#205,8)+#207+CN(#205,10)+#207+
       CN(#205,10)+#188;
END;

PROCEDURE SCR_4;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE SCRCAP4;
BEGIN
 CAP_T4; WRITELN (S0,'CALCULE VERIFICARE LA LUCRAREA ',
 NUMFIS);
 WRITELN (S0,S1); WRITELN (S0,S2); WRITELN (S0,S3);
 WRITELN (S0,S4);WRITELN (S0,S5); WRITELN (S0,S6);NL:=7;
END;
BEGIN
  CLRSCR;SCRCAP4;
  FOR I:=1 TO NTL DO
    BEGIN
      SEEK(D,I); READ (D,LD);
      WRITE (S0,#186,LD.CODS:6,#179,LD.CODV:6,#179,
             LD.DMR:10:4,#179);
      WRITELN (LD.P0:7:4,#179,LD.VI:8:4,#179,LD.DC:10:4,
               #179,LD.DD:10:4,#186);
      NL:=NL+1;
      IF NL>=NLE THEN IF I<NTL THEN
        BEGIN
          WRITELN (S0,S8);MESJ(60,WHEREY,MSG1);
          CLRSCR;SCRCAP4;
        END;
    END;
  WRITELN (S0,S8);MESJ(60,WHEREY,MSG1);
END;

PROCEDURE TIP_4;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE TIPCAP4;
```

```
 WRITELN(LST,S0,S6);NL:=7;
END;
BEGIN
  CLRSCR;TIPCAP4;
  FOR I:=1 TO NTL DO
    BEGIN
      SEEK(D,I); READ (D,LD);
      WRITE (LST,S0,#186,LD.CODS:6,#179,LD.CODV:6,#179,
             LD.DMR:10:4,#179);
      WRITELN (LST,LD.P0:7:4,#179,LD.VI:8:4,#179,LD.DC:10:4,
               #179,LD.DD:10:4,#186);
      NL:=NL+1;
      IF NL>=NLP THEN IF I<NTL THEN
        BEGIN
          WRITELN (LST,S0,S8);MESJ(60,WHEREY,MSG2);
          CLRSCR;TIPCAP4;
        END;
    END;
  WRITELN (LST,S0,S8);MESJ(60,WHEREY,MSG1);
END;

PROCEDURE CAP_T6;
BEGIN
  {    11     18            32            46            60
74
  S1:='|-----------------------------------------------------
---------|';
  S2:='|PUNCT |   COORDONATE PROVIZORII   |   COORDONATE
COMPENSATE   |';
  S3:='|      |---------------------------|-----------------
---------|';
  S4:='|      |   NORDp(m)  |   ESTp(m)   |     N(m)    |
E(m)     |';
  S5:='|-----------------------------------------------------
---------|';
  S7:='|      |             |             |             |
|';
  S8:='|-----------------------------------------------------
---------|';}
  S0:='          ';
  S1:=#201+CN(#205,6)+#209+CN(#205,27)+#209+CN(#205,27)+#187;
  S2:=#186+'PUNCT '+#179+'   COORDONATE PROVIZORII   '+#179+
      '   COORDONATE COMPENSATE   '+#186;
```

```
      CN(#205,13)+#216+CN(#205,13)+#185;
  S7:=#186+'      '+#179+'             '+#179+'             '
      +#179+'             '+#179+'             '+#186;
  S8:=#200+CN(#205,6)+#207+CN(#205,13)+#207+CN(#205,13)+#207+
      CN(#205,13)+#207+CN(#205,13)+#188;
END;

PROCEDURE SCR_6;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE SCRCAP6;
BEGIN
  CAP_T6; WRITELN (S0,'COORDONATE COMPENSATE LA LUCRAREA ',
  NUMFIS);
  WRITELN (S0,S1); WRITELN (S0,S2); WRITELN (S0,S3);
  WRITELN (S0,S4); WRITELN (S0,S5);NL:=5;
END;
BEGIN
  CLRSCR;SCRCAP6;
  FOR I:=1 TO NPT DO
    BEGIN
      SEEK(C,I); READ (C,LC);
      IF I<=NPV THEN
      WRITE (S0,#186,LC.CODP:6,#179,'      -      ',#179,
             '      -      ',#179)
      ELSE WRITE (S0,#186,LC.CODP:6,#179,LC.X:13:4,
                  #179,LC.Y:13:4,#179);
      WRITELN (LC.XC:13:4,#179,LC.YC:13:4,#186);NL:=NL+1;
      IF NL>=NLE THEN IF I<NPT THEN
        BEGIN
          WRITELN (S0,S8);MESJ(60,WHEREY,MSG1);
          CLRSCR;SCRCAP6;
        END;
    END;
  WRITELN (S0,S8);MESJ(60,WHEREY,MSG1);
END;

PROCEDURE TIP_6;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE TIPCAP6;
BEGIN
  CAP_T6; WRITELN (LST,S0,'COORDONATE COMPENSATE LA LUCRAREA
```

```
BEGIN
  MESJ(30,WHEREY,MSG2);TIPCAP6;
  FOR I:=1 TO NPT DO
    BEGIN
      SEEK(C,I); READ (C,LC);
      IF I<=NPV THEN
      WRITE (LST,S0,#186,LC.CODP:6,
             #179,'      -      ',#179,'      -      ',#179)
      ELSE
      WRITE
(LST,S0,#186,LC.CODP:6,#179,LC.X:13:4,#179,LC.Y:13:4,#179);
      WRITELN (LST,LC.XC:13:4,#179,LC.YC:13:4,#186);NL:=NL+1;
      IF NL>=NLP THEN
       IF I<NPT THEN
        BEGIN
          WRITELN
(LST,S0,S8);MESJ(30,WHEREY,MSG2);CLRSCR;TIPCAP6;
        END;
    END;
  WRITELN (LST,S0,S8);
END;

PROCEDURE CAP_T7;
BEGIN
  {    13     20        30        40        50        60
71
  S1:='-----------------------------------------------------
-----';
  S2:='|PUNCT | CORECTII COORD.PR.| ERORI COORD.COMP. |
EROARE  |';
  S3:='|      |---------|---------|---------|---------|
TOTALA  |';
  S4:='|      |  dN(m)  |  dE(m)  |   eN(m) |   eE(m) |
et(m)  |';
  S5:='|------|---------|---------|---------|---------|------
----|';
  S7:='|      |         |         |         |         |
|';
  S8:='|-----------------------------------------------------
----|';}
  S0:='             ';
  S1:=#201+CN(#205,6)+#209+CN(#205,19)+#209+CN(#205,19)+
      #209+CN(#205,10)+#187;
```

```
      '   eN(m) '+#179+'   eE(m) '+#179+'   et(m)  '+#186;
  S5:=#204+CN(#205,6)+#216+CN(#205,9)+#216+CN(#205,9)+#216+
      CN(#205,9)+#216+CN(#205,9)+#216+CN(#205,10)+#185;
  S7:=#186+'      '+#179+'         '+#179+'         '+
      #179+'         '+#179+'         '+#179+'
'+#186;
  S8:=#200+CN(#205,6)+#207+CN(#205,9)+#207+CN(#205,9)+
      #207+CN(#205,9)+#207+CN(#205,9)+#207+CN(#205,10)+#188;
END;

PROCEDURE SCR_7;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE SCRCAP7;
BEGIN
  CAP_T7; WRITELN (S0,'ERORI DE DETERMINARE LA LUCRAREA
',NUMFIS);
  WRITELN (S0,S1); WRITELN (S0,S2);
  WRITELN (S0,S3); WRITELN (S0,S4); WRITELN (S0,S5);NL:=5;
END;
BEGIN
  CLRSCR;SCRCAP7;
  FOR I:=NPV+1 TO NPT DO
    BEGIN
      SEEK(C,I); READ (C,LC);
      WRITELN
(S0,#186,LC.CODP:6,#179,LC.CX:9:4,#179,LC.CY:9:4,#179,
      LC.EX:9:6,#179,LC.EY:9:6,#179,LC.ET:10:6,#186);
      NL:=NL+1;
      IF NL>=NLE THEN
       IF I<NPT THEN
        BEGIN
          WRITELN
(S0,S8);MESJ(60,WHEREY,MSG1);CLRSCR;SCRCAP7;
        END;
    END;
    WRITELN (S0,S8);
    GOTOXY(1,WHEREY);SEEK(C,0); READ (C,LC);
    WRITELN(S0,'EROAREA MEDIE:');
    WRITELN (S0,'EMP UP=',LC.ET:10:6,' m');
    MESJ(60,WHEREY,MSG1);
END;
```

```
  CAP_T7; WRITELN (LST,S0,'ERORI DE DETERMINARE LA
  LUCRAREA ',NUMFIS);
  WRITELN (LST,S0,S1); WRITELN (LST,S0,S2);
  WRITELN (LST,S0,S3); WRITELN (LST,S0,S4);
   WRITELN(LST,S0,S5);NL:=5;
END;
BEGIN
  MESJ(30,WHEREY,MSG2);TIPCAP7;
  FOR I:=NPV+1 TO NPT DO
    BEGIN
      SEEK(C,I); READ (C,LC);
      WRITELN (LST,S0,#186,LC.CODP:6,#179,LC.CX:9:4,#179,
               LC.CY:9:4,#179,LC.EX:9:6,#179,LC.EY:9:6,#179,
               LC.ET:10:6,#186);
      NL:=NL+1;
      IF NL>=NLP THEN IF I<NPT THEN
        BEGIN
          WRITELN (LST,S0,S8);MESJ(30,WHEREY,MSG2);
          CLRSCR;TIPCAP7;
        END;
    END;
  WRITELN (LST,S0,S8);SEEK(C,0); READ (C,LC);
  WRITELN (LST,S0,'EROAREA MEDIE:');
  WRITELN (LST,S0,'EMP UP=',LC.ET:10:6,' m');
END;

PROCEDURE AFIS_M_E_IT(MSG:STRING;VAR E:FISR);
VAR S21,S22:STRING[6];
    S1,S2,S3:STRING[40];
    S11,S31,S41,S42:STRING[28];
    EE,EE1,EE2,SUMA:REAL;
    NP,NTCOL,POZ,I:LONGINT;
    LIN:INTEGER;
PROCEDURE cap1;
BEGIN
  WRITELN (s0,MSG);
  WRITELN (s0,s1,s11); WRITELN (s0,s2,s41,s21:6,s42);
  WRITELN (s0,s3,s31); WRITELN (s0,s1,s11);
END;
PROCEDURE cap2;
BEGIN
  WRITELN (' '); WRITELN (s0,'  -------',s11);
  WRITELN (s0,'  |NUMAR|',s41,s21:6,s42);
```

```
  WRITELN (' '); WRITELN (s0,'  -------',s11,s11);
  WRITELN (s0,'  |NUMAR|',s41,s21:6,s42,s41,s22:6,s42);
  WRITELN (s0,'  | EC. |',s31,s31);
  WRITELN (s0,'  -------',s11,s11);
END;
PROCEDURE sc(var el:real);
BEGIN
  IF ABS(el)>1E-6 THEN WRITE ('|',el:13:6)
  ELSE WRITE ('|      -      ');
END;
BEGIN
  S1:='----------------------------------------';
  S0:='        ';
  S2:='| PUNCT| PUNCT|NUMAR| PONDERE | TERMEN |';
  S3:='|STATIE| VIZAT| EC. |         |LIBER(m)|';
  S11:=COPY(S1,1,28);S41:='     PUNCTUL  ';S42:='       |';
  S31:='     dN      |     dE      |';
  NTCOL:=2*NPN+1;NP:=1;
  SEEK(C,(NPV+NP)); READ (C,LC);
  S21:=LC.CODP;CAP1;LIN:=5;
  FOR I:=1 TO NTL DO BEGIN
   SEEK(D,I); READ (D,LD);
   IF MSG<>'MATRICEA ECUATIILOR INITIALE' THEN LD.P0:=1.0;
   WRITE (S0,'|',LD.CODS:6,'|',LD.CODV:6,'|',I:5,'|',
          LD.P0:9:5);
   POZ:=I*NTCOL;SEEK(E,POZ); READ (E,EE);
   POZ:=(I-1)*NTCOL+2*NP-1;SEEK(E,POZ); READ (E,EE1);
   POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
   IF ABS(EE)>1E-5 THEN WRITE ('|',EE:8:4) ELSE
   WRITE ('|    -   ');
   SC(EE1);SC(EE2); WRITELN ('|');
   LIN:=LIN+1;
   IF (I=NTL) OR (LIN>NLE) THEN BEGIN
    WRITELN (S0,S1,S11);
    IF I<NTL THEN
    BEGIN MESJ(60,WHEREY,MSG1);CLRSCR;CAP1;LIN:=5; END;
   END;
   IF I=NTL THEN MESJ(60,WHEREY,MSG1);
  END;
  REPEAT
   NP:=NP+1;
   IF NP<=NPN THEN BEGIN
    MESJ(60,WHEREY,MSG1);
```

```
    END;
    LIN:=5;
    FOR I:=1 TO NTL DO BEGIN
     POZ:=(I-1)*NTCOL+2*NP-1;SEEK(E,POZ); READ (E,EE1);
     POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
     WRITE (S0,'  |',I:5);SC(EE1);SC(EE2);
     IF NP=NPN THEN WRITELN ('|');
     IF NP<NPN THEN BEGIN
      POZ:=POZ+1;SEEK(E,POZ); READ (E,EE1);
      POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
      SC(EE1);SC(EE2); WRITELN ('|');
     END;
     LIN:=LIN+1;
     IF (I=NTL) OR (LIN>NLE) THEN BEGIN
      WRITE (S0,'  -------',S11); IF NP=NPN THEN WRITELN (' 
');
      IF NP<NPN THEN WRITELN (S11);
      IF I<NTL THEN BEGIN
       MESJ(60,WHEREY,MSG1);
       IF NP=NPN THEN BEGIN CLRSCR;CAP2; END;
       IF NP<NPN THEN BEGIN CLRSCR;CAP3; END;
       LIN:=5;
      END;
      IF I=NTL THEN MESJ(60,WHEREY,MSG1);
     END;
    END;
    IF NP<NPN THEN NP:=NP+1;
   END;
  UNTIL NP=NPN;
END;

PROCEDURE TIP_M_E_IT(MSG:STRING;VAR E:FISR);
VAR S21,S22:STRING[6];
    S1,S2,S3:STRING[40];
    S11,S31,S41,S42:STRING[28];
    EE,EE1,EE2:REAL;
    NP,NTCOL,POZ,I:LONGINT;
    LIN:INTEGER;
PROCEDURE cap1;
BEGIN
  WRITELN (lst,s0,MSG);
  WRITELN (lst,s0,s1,s11); WRITELN (lst,s0,s2,s41,s21:6,s42);
  WRITELN (lst,s0,s3,s31); WRITELN (lst,s0,s1,s11);
```

```
  WRITELN (lst,s0,'  | EC. |',s31);
  WRITELN (lst,s0,'  -------',s11);
END;
PROCEDURE cap3;
BEGIN
  WRITELN (lst,' '); WRITELN (lst,s0,'  -------',s11,s11);
  WRITELN (lst,s0,'  |NUMAR|',s41,s21:6,s42,s41,s22:6,s42);
  WRITELN (lst,s0,'  | EC. |',s31,s31);
  WRITELN (lst,s0,'  -------',s11,s11);
END;
PROCEDURE sc(var el:real);
BEGIN
  IF ABS(el)>1E-6 THEN WRITE(LST,'|',el:13:6)
  ELSE WRITE (LST,'|      -      ');
END;
BEGIN
  S1:='--------------------------------------';S0:='
';
  S2:='| PUNCT| PUNCT|NUMAR| PONDERE | TERMEN |';
  S3:='|STATIE| VIZAT| EC. |         |LIBER cc|';
  S11:=COPY(S1,1,28);S41:='     PUNCTUL  ';S42:='       |';
  S31:='     dN      |     dE      |';
  NTCOL:=2*NPN+1;NP:=1;
  SEEK(C,(NPV+NP)); READ (C,LC);
  S21:=LC.CODP;
  MESJ(60,WHEREY,MSG1);TESTTIP;CAP1;LIN:=5;
  FOR I:=1 TO NTL DO BEGIN
   SEEK(D,I); READ (D,LD);
   IF MSG<>'MATRICEA ECUATIILOR INITIALE' THEN LD.P0:=1.0;
   WRITE
(LST,S0,'|',LD.CODS:6,'|',LD.CODV:6,'|',I:5,'|',LD.P0:9:5);
   POZ:=I*NTCOL;SEEK(E,POZ); READ (E,EE);
   POZ:=(I-1)*NTCOL+2*NP-1;SEEK(E,POZ); READ (E,EE1);
   POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
   IF ABS(EE)>1E-5 THEN WRITE (LST,'|',EE:8:4)
   ELSE WRITE (LST,'|    0   ');
   SC(EE1);SC(EE2); WRITELN (LST,'|');
   LIN:=LIN+1;
   IF (I=NTL) OR (LIN>NLP) THEN BEGIN
    WRITELN (LST,S0,S1,S11);
    IF I<NTL THEN BEGIN MESJ(30,WHEREY,MSG2);CAP1;LIN:=5;
END;
   END;
```

```
    SEEK(C,(NPV+NP)); READ (C,LC);S21:=LC.CODP;
    IF NP=NPN THEN BEGIN TESTTIP;CAP2; END;
    IF NP<NPN THEN BEGIN
     SEEK(C,(NPV+NP+1)); READ (C,LC);
     S22:=LC.CODP;TESTTIP;CAP3;
    END;
    LIN:=5;
    FOR I:=1 TO NTL DO BEGIN
     POZ:=(I-1)*NTCOL+2*NP-1;SEEK(E,POZ); READ (E,EE1);
     POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
     WRITE (LST,S0,'  |',I:5);SC(EE1);SC(EE2);
     IF NP=NPN THEN WRITELN (LST,'|');
     IF NP<NPN THEN BEGIN
      POZ:=POZ+1;SEEK(E,POZ); READ (E,EE1);
      POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
      SC(EE1);SC(EE2); WRITELN (LST,'|');
     END;
     LIN:=LIN+1;
     IF (I=NTL) OR (LIN>NLP) THEN BEGIN
      WRITE (LST,S0,'  -------',S11);
      IF NP=NPN THEN WRITELN (LST,' ');
      IF NP<NPN THEN WRITELN (LST,S11);
      IF I<NTL THEN BEGIN
       MESJ(30,WHEREY,MSG2);TESTTIP;
       IF NP=NPN THEN CAP2;
       IF NP<NPN THEN CAP3;
       LIN:=5;
      END;
     END;
    END;
    IF NP<NPN THEN NP:=NP+1;
   END;
  UNTIL NP=NPN;
END;

PROCEDURE Text_M_E_IT(MSG:STRING;VAR E:FISR;VAR F:Text);
VAR S21,S22:STRING[6];
    S1,S2,S3:STRING[40];
    S11,S31,S41,S42:STRING[28];
    EE,EE1,EE2:REAL;
    NP,NTCOL,POZ,I:LONGINT;
    LIN:INTEGER;
PROCEDURE cap1;
```

```
PROCEDURE cap2;
BEGIN
  WRITELN (F,' '); WRITELN (F,s0,'  -------',s11);
  WRITELN (F,s0,'  |NUMAR|',s41,s21:6,s42);
  WRITELN (F,s0,'  | EC. |',s31);
  WRITELN (F,s0,'  -------',s11);
END;
PROCEDURE cap3;
BEGIN
  WRITELN (F,' '); WRITELN (F,s0,'  -------',s11,s11);
  WRITELN (F,s0,'  |NUMAR|',s41,s21:6,s42,s41,s22:6,s42);
  WRITELN (F,s0,'  | EC. |',s31,s31);
  WRITELN (F,s0,'  -------',s11,s11);
END;
PROCEDURE sc(var el:real);
BEGIN
  IF ABS(el)>1E-6 THEN WRITE (F,'|',el:13:6)
  ELSE WRITE (F,'|      -      ');
END;
BEGIN
  S1:='----------------------------------------';S0:='
';
  S2:='| PUNCT| PUNCT|NUMAR| PONDERE | TERMEN |';
  S3:='|STATIE| VIZAT| EC. |         |LIBER cc|';
  S11:=COPY(S1,1,28);S41:='     PUNCTUL  ';S42:='       |';
  S31:='     dN      |     dE      |';
  NTCOL:=2*NPN+1;NP:=1;
  SEEK(C,(NPV+NP)); READ (C,LC);
  S21:=LC.CODP; WRITELN (F,' ');CAP1;LIN:=5;
  FOR I:=1 TO NTL DO BEGIN
   SEEK(D,I); READ (D,LD);
   IF MSG<>'MATRICEA ECUATIILOR INITIALE' THEN LD.P0:=1.0;
   WRITE
(F,S0,'|',LD.CODS:6,'|',LD.CODV:6,'|',I:5,'|',LD.P0:9:5);
   POZ:=I*NTCOL;SEEK(E,POZ); READ (E,EE);
   POZ:=(I-1)*NTCOL+2*NP-1;SEEK(E,POZ); READ (E,EE1);
   POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
   IF ABS(EE)>1E-5 THEN WRITE (F,'|',EE:8:4)
   ELSE WRITE (F,'|    0   ');
   SC(EE1);SC(EE2); WRITELN (F,'|');
   LIN:=LIN+1;
   IF (I=NTL) OR (LIN>NLP) THEN BEGIN
    WRITELN (F,S0,S1,S11);
```

```
   IF NP<=NPN THEN BEGIN
    SEEK(C,(NPV+NP)); READ (C,LC);S21:=LC.CODP;
    IF NP=NPN THEN CAP2;
    IF NP<NPN THEN BEGIN
     SEEK(C,(NPV+NP+1)); READ (C,LC);
     S22:=LC.CODP; WRITELN (F,' ');CAP3;
    END;
    LIN:=5;
    FOR I:=1 TO NTL DO BEGIN
     POZ:=(I-1)*NTCOL+2*NP-1;SEEK(E,POZ); READ (E,EE1);
     POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
     WRITE (F,S0,'  |',I:5);SC(EE1);SC(EE2);
     IF NP=NPN THEN WRITELN (F,'|');
     IF NP<NPN THEN BEGIN
      POZ:=POZ+1;SEEK(E,POZ); READ (E,EE1);
      POZ:=POZ+1;SEEK(E,POZ); READ (E,EE2);
      SC(EE1);SC(EE2); WRITELN (F,'|');
     END;
     LIN:=LIN+1;
     IF (I=NTL) OR (LIN>NLP) THEN BEGIN
      WRITE (F,S0,'  -------',S11);
      IF NP=NPN THEN WRITELN (F,' ');
      IF NP<NPN THEN WRITELN (F,S11);
      IF I<NTL THEN BEGIN
       WRITELN (F,' ');
       IF NP=NPN THEN CAP2;
       IF NP<NPN THEN CAP3;
       LIN:=5;
      END;
     END;
    END;
    IF NP<NPN THEN NP:=NP+1;
   END;
  UNTIL NP=NPN;
END;

PROCEDURE TEXT_3;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE TEXTCAP3;
BEGIN
 CAP_T3; WRITELN (F,S0,'CALCULE COMPENSARE LA LUCRAREA ',
 NUMFIS);
```

```
 TEXTCAP3;
 FOR I:=1 TO NTL DO BEGIN
  SEEK(D,I); READ (D,LD);
  WRITE (F,S0,#186,LD.CODS:6,#179,LD.CODV:6,#179,
         LD.D0:10:4,#179);
  WRITELN (F,LD.T0:9:4,#179,LD.A0:9:6,#179,LD.B0:9:6,#179,
           LD.L0:8:4,#186);NL:=NL+1;
  IF NL>=NLP THEN IF I<NTL THEN BEGIN
   WRITELN (F,S0,S8);TEXTCAP3;
  END;
 END;
 WRITELN (F,S0,S8); WRITELN (F,S0);
END;

PROCEDURE TEXT_4;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE TEXTCAP4;
BEGIN
 CAP_T4; WRITELN (F,S0,'CALCULE VERIFICARE LA LUCRAREA ',
 NUMFIS);
 WRITELN (F,S0,S1); WRITELN (F,S0,S2); WRITELN (F,S0,S3);
 WRITELN (F,S0,S4); WRITELN (F,S0,S5); WRITELN (F,S0,S6);
 NL:=7;
END;
BEGIN
 TEXTCAP4;
 FOR I:=1 TO NTL DO BEGIN
  SEEK(D,I); READ (D,LD);
  WRITE (F,S0,#186,LD.CODS:6,#179,LD.CODV:6,#179,
         LD.DMR:10:4,#179);
  WRITELN (F,LD.P0:7:4,#179,LD.VI:8:4,#179,LD.DC:10:4,#179,
           LD.DD:10:4,#186);
  NL:=NL+1;
  IF NL>=NLP THEN IF I<NTL THEN BEGIN
   WRITELN (F,S0,S8);TEXTCAP4;
  END;
 END;
 WRITELN (F,S0,S8); WRITELN (F,S0);
END;

PROCEDURE TEXT_6;
VAR NL:INTEGER;
```

```
  WRITELN (F,S0,S1); WRITELN (F,S0,S2); WRITELN (F,S0,S3);
  WRITELN (F,S0,S4); WRITELN (F,S0,S5);NL:=5;
END;
BEGIN
 TEXTCAP6;
 FOR I:=1 TO NPT DO BEGIN
  SEEK(C,I); READ (C,LC);
  IF I<=NPV THEN WRITE (F,S0,#186,LC.CODP:6,#179,
                 '      -      ',#179,'      -      ',#179)
  ELSE WRITE (F,S0,#186,LC.CODP:6,#179,LC.X:13:4,#179,
              LC.Y:13:4,#179);
  WRITELN (F,LC.XC:13:4,#179,LC.YC:13:4,#186);NL:=NL+1;
  IF NL>=NLP THEN IF I<NPT THEN BEGIN
   WRITELN (F,S0,S8);TEXTCAP6;
  END;
 END;
 WRITELN (F,S0,S8); WRITELN (F,S0);
END;

PROCEDURE TEXT_7;
VAR NL:INTEGER;
    I:LONGINT;
PROCEDURE TEXTCAP7;
BEGIN
  CAP_T7; WRITELN (F,S0,'ERORI DE DETERMINARE LA LUCRAREA ',
  NUMFIS);
  WRITELN (F,S0,S1); WRITELN (F,S0,S2);
  WRITELN (F,S0,S3); WRITELN (F,S0,S4); WRITELN (F,S0,S5);
  NL:=5;
END;
BEGIN
 TEXTCAP7;
 FOR I:=NPV+1 TO NPT DO BEGIN
  SEEK(C,I); READ (C,LC);
  WRITELN (F,S0,#186,LC.CODP:6,#179,LC.CX:9:4,#179,
           LC.CY:9:4,#179,LC.EX:9:6,#179,LC.EY:9:6,#179,
           LC.ET:10:6,#186);NL:=NL+1;
  IF NL>=NLP THEN IF I<NPT THEN BEGIN
   WRITELN (F,S0,S8);TEXTCAP7;
  END;
 END;
 WRITELN (F,S0,S8);SEEK(C,0); READ (C,LC);
 WRITELN (F,S0,'EROAREA MEDIE:');
```

5.3 Exemplu de calcul rezolvat cu programul prezentat

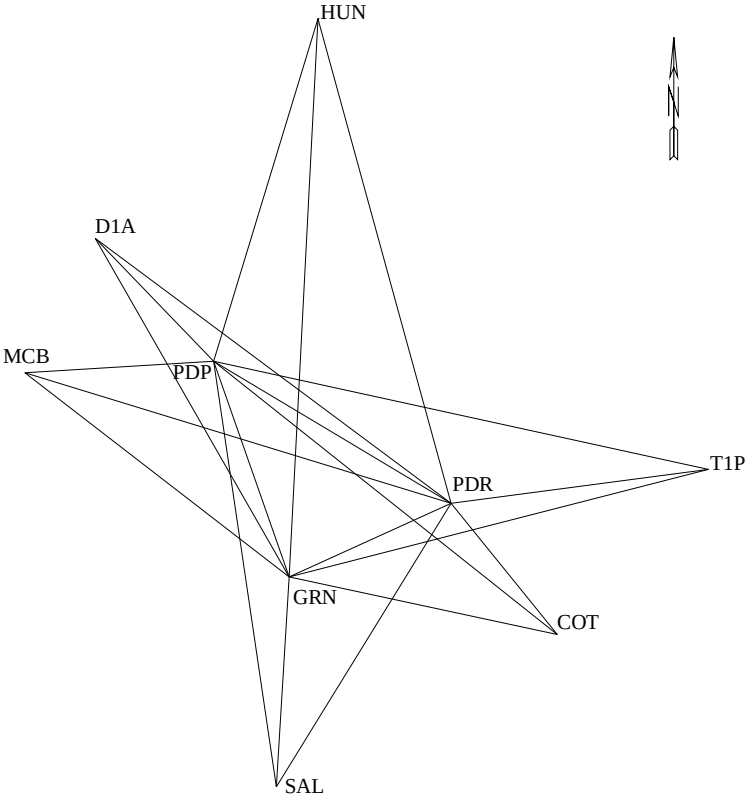

Fig. 5.2 Schița rețelei de trilaterație din exemplul de calcul

CALCULE COMPENSARE LA LUCRAREA Exemplu Calcul

LATURA RETEA		DISTANTA	ORIENTARE	COEF. ECUATII		TERMEN
DE LA PCT.	LA PCT.	PROVIZ. (m)	PROVIZ. (gon)	a	b	LIBER (m)
HUN	PDR	1750.1508	182.8391	-0.963887	0.266311	-0.0002
HUN	GRN	1945.9981	203.3247	-0.998637	-0.052200	0.0001
HUN	PDP	1247.2792	218.9735	-0.955915	-0.293643	0.0002
D1A	PDR	1552.0936	140.5096	-0.594243	0.804286	-0.0004
D1A	PDP	596.6099	150.8811	-0.716825	0.697253	-0.0001
D1A	GRN	1361.1294	166.6615	-0.865985	0.500071	0.0004
MCB	PDP	664.2127	96.0754	0.061608	0.998100	-0.0003
MCB	PDR	1562.6153	118.7556	-0.290368	0.956915	0.0003
MCB	GRN	1168.2347	141.5949	-0.607867	0.794039	-0.0003
SAL	PDP	1500.6528	390.6240	0.989174	-0.146746	-0.0002
SAL	GRN	734.7013	3.8546	0.998167	0.060512	0.0003
SAL	PDR	1163.7460	35.2614	0.850488	0.525994	0.0000
COT	GRN	960.7182	313.5082	0.210598	-0.977573	0.0002
COT	PDP	1535.6336	342.6414	0.620836	-0.783940	-0.0004
COT	PDR	590.2903	356.6633	0.777113	-0.629362	0.0003
T1P	GRN	1515.5658	284.0866	-0.247372	-0.968921	-0.0002
T1P	PDR	908.5593	291.6723	-0.130438	-0.991456	0.0003
T1P	PDP	1773.4838	313.6056	0.212093	-0.977249	-0.0002
PDP	HUN	1247.2792	18.9735	0.955915	0.293643	0.0022
PDP	T1P	1773.4838	113.6056	-0.212093	0.977249	-0.0012
PDP	PDR	968.2313	134.1365	-0.510885	0.859649	0.0003
PDP	COT	1535.6336	142.6414	-0.620836	0.783940	0.0006
PDP	GRN	796.3234	178.4304	-0.943150	0.332369	0.0004
PDP	SAL	1500.6528	190.6240	-0.989174	0.146746	0.0008
PDP	MCB	664.2127	296.0754	-0.061608	-0.998100	-0.0013
PDP	D1A	596.6099	350.8811	0.716825	-0.697253	-0.0011
PDR	HUN	1750.1508	382.8391	0.963887	-0.266311	0.0018
PDR	T1P	908.5593	91.6723	0.130438	0.991456	-0.0007
PDR	COT	590.2903	156.6633	-0.777113	0.629362	0.0013
PDR	SAL	1163.7460	235.2614	-0.850488	-0.525994	0.0020
PDR	GRN	622.8837	272.9920	-0.411629	-0.911352	-0.0013
PDR	MCB	1562.6153	318.7556	0.290368	-0.956915	-0.0007
PDR	PDP	968.2313	334.1365	0.510885	-0.859649	0.0013
PDR	D1A	1552.0936	340.5096	0.594243	-0.804286	0.0006
GRN	HUN	1945.9981	3.3247	0.998637	0.052200	-0.0009
GRN	PDR	622.8837	72.9920	0.411629	0.911352	-0.0013
GRN	T1P	1515.5658	84.0866	0.247372	0.968921	-0.0012
GRN	COT	960.7182	113.5082	-0.210598	0.977573	0.0012
GRN	SAL	734.7013	203.8546	-0.998167	-0.060512	0.0003
GRN	MCB	1168.2347	341.5949	0.607867	-0.794039	0.0007
GRN	D1A	1361.1294	366.6615	0.865985	-0.500071	0.0004
GRN	PDP	796.3234	378.4304	0.943150	-0.332369	0.0004

CALCULE VERIFICARE LA LUCRAREA Exemplu Calcul

LATURA RETEA		DISTANTE	PONDERE	CORECTII	DISTANTE	DISTANTE
DE LA PCT.	LA PCT.	MASURATE (m)	MASU-RARE	DISTANTE MAS. (m)	COMPENS. (m)	DIN COORD.COMP (m)
HUN	PDR	1750.1510	0.3373	0.0001	1750.1511	1750.1511
HUN	GRN	1945.9980	0.3033	0.0005	1945.9985	1945.9985
HUN	PDP	1247.2790	0.4733	-0.0001	1247.2789	1247.2789
D1A	PDR	1552.0940	0.3803	-0.0002	1552.0938	1552.0938
D1A	PDP	596.6100	0.9894	0.0003	596.6103	596.6103
D1A	GRN	1361.1290	0.4337	0.0004	1361.1294	1361.1294
MCB	PDP	664.2130	0.8887	0.0005	664.2135	664.2135
MCB	PDR	1562.6150	0.3778	0.0005	1562.6155	1562.6155
MCB	GRN	1168.2350	0.5053	-0.0005	1168.2345	1168.2345
SAL	PDP	1500.6530	0.3934	-0.0002	1500.6528	1500.6528
SAL	GRN	734.7010	0.8034	-0.0000	734.7010	734.7010
SAL	PDR	1163.7460	0.5072	-0.0001	1163.7459	1163.7459
COT	GRN	960.7180	0.6144	0.0007	960.7187	960.7187
COT	PDP	1535.6340	0.3844	-0.0009	1535.6331	1535.6331
COT	PDR	590.2900	1.0000	-0.0000	590.2900	590.2900
T1P	GRN	1515.5660	0.3895	0.0004	1515.5664	1515.5664
T1P	PDR	908.5590	0.6497	0.0002	908.5592	908.5592
T1P	PDP	1773.4840	0.3328	-0.0009	1773.4831	1773.4831
PDP	HUN	1247.2770	0.4733	0.0019	1247.2789	1247.2789
PDP	T1P	1773.4850	0.3328	-0.0019	1773.4831	1773.4831
PDP	PDR	968.2310	0.6097	-0.0001	968.2309	968.2309
PDP	COT	1535.6330	0.3844	0.0001	1535.6331	1535.6331
PDP	GRN	796.3230	0.7413	0.0003	796.3233	796.3233
PDP	SAL	1500.6520	0.3934	0.0008	1500.6528	1500.6528
PDP	MCB	664.2140	0.8887	-0.0005	664.2135	664.2135
PDP	D1A	596.6110	0.9894	-0.0007	596.6103	596.6103
PDR	HUN	1750.1490	0.3373	0.0021	1750.1511	1750.1511
PDR	T1P	908.5600	0.6497	-0.0008	908.5592	908.5592
PDR	COT	590.2890	1.0000	0.0010	590.2900	590.2900
PDR	SAL	1163.7440	0.5072	0.0019	1163.7459	1163.7459
PDR	GRN	622.8850	0.9477	-0.0006	622.8844	622.8844
PDR	MCB	1562.6160	0.3778	-0.0005	1562.6155	1562.6155
PDR	PDP	968.2300	0.6097	0.0009	968.2309	968.2309
PDR	D1A	1552.0930	0.3803	0.0008	1552.0938	1552.0938
GRN	HUN	1945.9990	0.3033	-0.0005	1945.9985	1945.9985
GRN	PDR	622.8850	0.9477	-0.0006	622.8844	622.8844
GRN	T1P	1515.5670	0.3895	-0.0006	1515.5664	1515.5664
GRN	COT	960.7170	0.6144	0.0017	960.7187	960.7187
GRN	SAL	734.7010	0.8034	-0.0000	734.7010	734.7010
GRN	MCB	1168.2340	0.5053	0.0005	1168.2345	1168.2345
GRN	D1A	1361.1290	0.4337	0.0004	1361.1294	1361.1294
GRN	PDP	796.3230	0.7413	0.0003	796.3233	796.3233

COORDONATE COMPENSATE LA LUCRAREA Exemplu Calcul

PUNCT	COORDONATE PROVIZORII		COORDONATE COMPENSATE	
	NORDp(m)	ESTp(m)	N(m)	E(m)
HUN	-	-	10278.6000	9326.4000
D1A	-	-	9513.9720	8544.1570
MCB	-	-	9045.3860	8297.1940
SAL	-	-	7601.9000	9180.3600
COT	-	-	8132.9300	10163.9900
T1P	-	-	8710.1630	10693.2810
GRN	8335.2550	9224.8180	8335.2547	9224.8174
PDR	8591.6520	9792.4840	8591.6518	9792.4841
PDP	9086.3070	8960.1450	9086.3071	8960.1457

ERORI DE DETERMINARE LA LUCRAREA Exemplu Calcul

PUNCT	CORECTII COORD.PR.		ERORI COORD.COMP.		EROARE TOTALA
	dN(m)	dE(m)	eN(m)	eE(m)	et(m)
GRN	-0.0003	-0.0006	0.000280	0.000296	0.000407
PDR	-0.0002	0.0001	0.000338	0.000282	0.000440
PDP	0.0001	0.0007	0.000305	0.000319	0.000442

EROAREA MEDIE:
EMP UP= 0.000624 m

MATRICEA ECUATIILOR INITIALE

PUNCT STATIE	PUNCT VIZAT	NUMAR EC.	PONDERE	TERMEN LIBER cc	PUNCTUL dN	GRN dE
HUN	PDR	1	0.33728	-0.0002	-	-
HUN	GRN	2	0.30333	0.0001	-0.998637	-0.052200
HUN	PDP	3	0.47326	0.0002	-	-
D1A	PDR	4	0.38032	-0.0004	-	-
D1A	PDP	5	0.98941	-0.0001	-	-
D1A	GRN	6	0.43368	0.0004	-0.865985	0.500071
MCB	PDP	7	0.88870	-0.0003	-	-
MCB	PDR	8	0.37776	0.0003	-	-
MCB	GRN	9	0.50528	-0.0003	-0.607867	0.794039
SAL	PDP	10	0.39335	-0.0002	-	-
SAL	GRN	11	0.80344	0.0003	0.998167	0.060512
SAL	PDR	12	0.50723	0.0000	-	-
COT	GRN	13	0.61442	0.0002	0.210598	-0.977573
COT	PDP	14	0.38439	-0.0004	-	-
COT	PDR	15	1.00000	0.0003	-	-
T1P	GRN	16	0.38948	-0.0002	-0.247372	-0.968921
T1P	PDR	17	0.64970	0.0003	-	-
T1P	PDP	18	0.33284	-0.0002	-	-
PDP	HUN	19	0.47326	0.0022	-	-
PDP	T1P	20	0.33284	-0.0012	-	-
PDP	PDR	21	0.60966	0.0003	-	-
PDP	COT	22	0.38439	0.0006	-	-
PDP	GRN	23	0.74127	0.0004	-0.943150	0.332369
PDP	SAL	24	0.39336	0.0008	-	-
PDP	MCB	25	0.88870	-0.0013	-	-
PDP	D1A	26	0.98940	-0.0011	-	-
PDR	HUN	27	0.33728	0.0018	-	-
PDR	T1P	28	0.64970	-0.0007	-	-
PDR	COT	29	1.00000	0.0013	-	-
PDR	SAL	30	0.50723	0.0020	-	-
PDR	GRN	31	0.94767	-0.0013	-0.411629	-0.911352
PDR	MCB	32	0.37776	-0.0007	-	-
PDR	PDP	33	0.60966	0.0013	-	-
PDR	D1A	34	0.38032	0.0006	-	-
GRN	HUN	35	0.30333	-0.0009	-0.998637	-0.052200
GRN	PDR	36	0.94767	-0.0013	-0.411629	-0.911352
GRN	T1P	37	0.38948	-0.0012	-0.247372	-0.968921
GRN	COT	38	0.61443	0.0012	0.210598	-0.977573
GRN	SAL	39	0.80344	0.0003	0.998167	0.060512
GRN	MCB	40	0.50528	0.0007	-0.607867	0.794039
GRN	D1A	41	0.43368	0.0004	-0.865985	0.500071
GRN	PDP	42	0.74127	0.0004	-0.943150	0.332369

MATRICEA ECUATIILOR INITIALE-continuare

NUMAR EC.	PUNCTUL dN	PDR dE	PUNCTUL dN	PDP dE
1	-0.963887	0.266311	-	-
2	-	-	-	-
3	-	-	-0.955915	-0.293643
4	-0.594243	0.804286	-	-
5	-	-	-0.716825	0.697253
6	-	-	-	-
7	-	-	0.061608	0.998100
8	-0.290368	0.956915	-	-
9	-	-	-	-
10	-	-	0.989174	-0.146746
11	-	-	-	-
12	0.850488	0.525994	-	-
13	-	-	-	-
14	-	-	0.620836	-0.783940
15	0.777113	-0.629362	-	-
16	-	-	-	-
17	-0.130438	-0.991456	-	-
18	-	-	0.212093	-0.977249
19	-	-	-0.955915	-0.293643
20	-	-	0.212093	-0.977249
21	-0.510885	0.859649	0.510885	-0.859649
22	-	-	0.620836	-0.783940
23	-	-	0.943150	-0.332369
24	-	-	0.989174	-0.146746
25	-	-	0.061608	0.998100
26	-	-	-0.716825	0.697253
27	-0.963887	0.266311	-	-
28	-0.130438	-0.991456	-	-
29	0.777113	-0.629362	-	-
30	0.850488	0.525994	-	-
31	0.411629	0.911352	-	-
32	-0.290368	0.956915	-	-
33	-0.510885	0.859649	0.510885	-0.859649
34	-0.594243	0.804286	-	-
35	-	-	-	-
36	0.411629	0.911352	-	-
37	-	-	-	-
38	-	-	-	-
39	-	-	-	-
40	-	-	-	-
41	-	-	-	-
42	-	-	0.943150	-0.332369

MATRICEA ECUATIILOR TRANSFORMATE

PUNCT STATIE	PUNCT VIZAT	NUMAR EC.	PONDERE	TERMEN LIBER cc	PUNCTUL dN	GRN dE
HUN	PDR	1	1.00000	-0.0001	-	-
HUN	GRN	2	1.00000	0.0001	-0.550008	-0.028750
HUN	PDP	3	1.00000	0.0001	-	-
D1A	PDR	4	1.00000	-0.0003	-	-
D1A	PDP	5	1.00000	-0.0001	-	-
D1A	GRN	6	1.00000	0.0002	-0.570286	0.329317
MCB	PDP	7	1.00000	-0.0003	-	-
MCB	PDR	8	1.00000	0.0002	-	-
MCB	GRN	9	1.00000	-0.0002	-0.432091	0.564429
SAL	PDP	10	1.00000	-0.0001	-	-
SAL	GRN	11	1.00000	0.0003	0.894706	0.054240
SAL	PDR	12	1.00000	0.0000	-	-
COT	GRN	13	1.00000	0.0002	0.165078	-0.766273
COT	PDP	14	1.00000	-0.0003	-	-
COT	PDR	15	1.00000	0.0003	-	-
T1P	GRN	16	1.00000	-0.0001	-0.154381	-0.604690
T1P	PDR	17	1.00000	0.0003	-	-
T1P	PDP	18	1.00000	-0.0001	-	-
PDP	HUN	19	1.00000	0.0015	-	-
PDP	T1P	20	1.00000	-0.0007	-	-
PDP	PDR	21	1.00000	0.0002	-	-
PDP	COT	22	1.00000	0.0004	-	-
PDP	GRN	23	1.00000	0.0003	-0.812023	0.286159
PDP	SAL	24	1.00000	0.0005	-	-
PDP	MCB	25	1.00000	-0.0012	-	-
PDP	D1A	26	1.00000	-0.0011	-	-
PDR	HUN	27	1.00000	0.0010	-	-
PDR	T1P	28	1.00000	-0.0005	-	-
PDR	COT	29	1.00000	0.0013	-	-
PDR	SAL	30	1.00000	0.0014	-	-
PDR	GRN	31	1.00000	-0.0013	-0.400714	-0.887185
PDR	MCB	32	1.00000	-0.0004	-	-
PDR	PDP	33	1.00000	0.0010	-	-
PDR	D1A	34	1.00000	0.0004	-	-
GRN	HUN	35	1.00000	-0.0005	-0.550007	-0.028750
GRN	PDR	36	1.00000	-0.0013	-0.400714	-0.887185
GRN	T1P	37	1.00000	-0.0008	-0.154381	-0.604690
GRN	COT	38	1.00000	0.0009	0.165078	-0.766273
GRN	SAL	39	1.00000	0.0003	0.894706	0.054240
GRN	MCB	40	1.00000	0.0005	-0.432092	0.564429
GRN	D1A	41	1.00000	0.0002	-0.570286	0.329317
GRN	PDP	42	1.00000	0.0003	-0.812023	0.286159

MATRICEA ECUATIILOR TRANSFORMATE-continuare

NUMAR EC.	PUNCTUL dN	PDR dE	PUNCTUL dN	PDP dE
1	-0.559784	0.154662	-	-
2	-	-	-	-
3	-	-	-0.657612	-0.202009
4	-0.366469	0.496002	-	-
5	-	-	-0.713018	0.693549
6	-	-	-	-
7	-	-	0.058079	0.940920
8	-0.178466	0.588139	-	-
9	-	-	-	-
10	-	-	0.620390	-0.092036
11	-	-	-	-
12	0.605719	0.374614	-	-
13	-	-	-	-
14	-	-	0.384916	-0.486039
15	0.777112	-0.629361	-	-
16	-	-	-	-
17	-0.105138	-0.799152	-	-
18	-	-	0.122362	-0.563799
19	-	-	-0.657613	-0.202009
20	-	-	0.122362	-0.563799
21	-0.398902	0.671219	0.398902	-0.671219
22	-	-	0.384916	-0.486040
23	-	-	0.812023	-0.286159
24	-	-	0.620390	-0.092036
25	-	-	0.058079	0.940920
26	-	-	-0.713017	0.693549
27	-0.559785	0.154662	-	-
28	-0.105138	-0.799152	-	-
29	0.777113	-0.629362	-	-
30	0.605720	0.374615	-	-
31	0.400714	0.887185	-	-
32	-0.178466	0.588138	-	-
33	-0.398902	0.671219	0.398902	-0.671219
34	-0.366469	0.496003	-	-
35	-	-	-	-
36	0.400714	0.887185	-	-
37	-	-	-	-
38	-	-	-	-
39	-	-	-	-
40	-	-	-	-
41	-	-	-	-
42	-	-	0.812023	-0.286159

SISTEMUL DE ECUATII NORMALE
MATRICE COEFICIENTI

		C 1	C 2	C 3	C 4
L	1	4.971947	-0.554702	-0.321143	-0.711015
L	2	-0.554702	4.505216	-0.711015	-1.574195
L	3	-0.321143	-0.711015	3.562113	-0.927409
L	4	-0.711015	-1.574195	-0.927409	6.057107
L	5	-1.318762	0.464736	-0.318246	0.535501
L	6	0.464736	-0.163774	0.535501	-0.901069

		C 5	C 6
L	1	-1.318762	0.464736
L	2	0.464736	-0.163774
L	3	-0.318246	0.535501
L	4	0.535501	-0.901069
L	5	4.621484	-2.240620
L	6	-2.240620	5.004289

VECTOR TERMENI LIBERI

T	1=	-0.001203
T	2=	-0.002507
T	3=	-0.000064
T	4=	0.001559
T	5=	-0.000862
T	6=	0.003106

SISTEMUL DE ECUATII REDUS
MATRICE REDUSA

		C 1	C 2	C 3	C 4
L	1	4.971947	-0.554702	-0.321143	-0.711015
L	2	-	4.443330	-0.746844	-1.653521
L	3	-	-	3.415839	-1.251261
L	4	-	-	-	4.881743
L	5	-	-	-	-
L	6	-	-	-	-

		C 5	C 6
L	1	-1.318762	0.464736
L	2	0.317606	-0.111925
L	3	-0.350042	0.546706
L	4	0.336879	-0.675996
L	5	4.189874	-2.006679
L	6	-	3.815851

VECTOR REDUS

T	1=	-0.001203
T	2=	-0.002641
T	3=	-0.000586
T	4=	0.000189
T	5=	-0.001066
T	6=	0.002762

MATRICEA COEFICIENTILOR DE PONDERE

		C 1	C 2	C 3	C 4
L	1	0.201128	0.025109	0.024399	0.044052
L	2	-	0.225056	0.049207	0.088842
L	3	-	-	0.292754	0.075037
L	4	-	-	-	0.204845
L	5	-	-	-	-
L	6	-	-	-	-

		C 5	C 6
L	1	0.059898	0.012049
L	2	-0.020092	0.004724
L	3	0.018425	-0.018961
L	4	-0.016470	0.027628
L	5	0.238671	0.125512

Bibliografie

1. Cristea V. ş. a. –Borland Pascal 7.0 pentru Windows
Editura Teora , 1994

2. Ghițău D. -Geodezie şi gravimetrie geodezică
E.D.P. Bucureşti 1983

3. Nistor Gh. - Teoria prelucrării măsurătorilor geodezice
Univ.Tehnică Iaşi - 1996

4. ** - Manualul inginerului geodez(vol. I, II, III)**
Editura Tehnică, Bucureşti, 1972-1974

5. ** - Rec Elta – Operating Instructions**
Carl Zeiss Oberkochen, Germany, 1991